Oracle® Linux

Administrator's Guide for Release 7

ORACLE

E54669-45
February 2017

Oracle Legal Notices

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Abstract

This manual provides an introduction to administering various features of Oracle Linux 7 systems.

Document generated on: 2017-02-28 (revision: 4019)

Table of Contents

Preface

The *Oracle Linux Administrator's Guide* provides introductory information about administering various features of Oracle Linux 7 systems, including system configuration, networking, network services, storage devices, file systems, authentication, and security.

Audience

This document is intended for administrators who need to configure and administer Oracle Linux. It is assumed that readers are familiar with web technologies and have a general understanding of using the Linux operating system, including knowledge of how to use a text editor such as `emacs` or `vim`, essential commands such as `cd`, `chmod`, `chown`, `ls`, `mkdir`, `mv`, `ps`, `pwd`, and `rm`, and using the `man` command to view manual pages.

Document Organization

The document is organized as follows:

- Part I, "System Configuration" describes how to configure software and kernel updates, booting, kernel and module settings, and devices, how to schedule tasks, and how to monitor and tune your system.

- Part II, "Networking and Network Services" describes how to configure network interfaces, network addresses, name service, network time services, basic web and email services, load balancing, and high availability.

- Part III, "Storage and File Systems" describes how to configure storage devices and how to create and manage local, shared, and cluster file systems.

- Part IV, "Authentication and Security" describes how to configure user account databases and authentication, how to add group and user accounts, how to administer essential aspects of system security, and how to configure and use the OpenSSH tools.

- Part V, "Containers" describes how to configure containers to isolate applications from the other processes that are running on a host system.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

The documentation for this product is available at:

http://www.oracle.com/technetwork/server-storage/linux/documentation/index.html.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Part I System Configuration

This section contains the following chapters:

- Chapter 1, *The Unbreakable Linux Network* describes how to access and use the software channels that are available on the Unbreakable Linux Network (ULN).

- Chapter 2, *Yum* describes how you can use the `yum` utility to install and upgrade software packages.

- Chapter 3, *Ksplice Uptrack* describes how to configure Ksplice Uptrack to update the kernel on a running system.

- Chapter 4, *Boot and Service Configuration* describes the Oracle Linux boot process, how to use the GRUB boot loader, how to change the run level of a system, and how to configure the services that are available at each run level.

- Chapter 5, *System Configuration Settings* describes the files and virtual file systems that you can use to change configuration settings for your system.

- Chapter 6, *Kernel Modules* describes how to load, unload, and modify the behavior of kernel modules.

- Chapter 7, *Device Management* describes how the system uses device files and how the udev device manager dynamically creates or removes device node files.

- Chapter 8, *Task Management* describes how to configure the system to run tasks automatically within a specific period of time, at a specified time and date, or when the system is lightly loaded.

- Chapter 9, *System Monitoring and Tuning* describes how to collect diagnostic information about a system for Oracle Support, and how to monitor and tune the performance of a system.

- Chapter 10, *System Dump Analysis* describes how to configure a system to create a memory image in the event of a system crash, and how to use the `crash` debugger to analyse the memory image in a crash dump or for a live system.

Table of Contents

Chapter 1 The Unbreakable Linux Network

Table of Contents

This chapter describes how to access and use the software channels that are available on the Unbreakable Linux Network (ULN).

1.1 About the Unbreakable Linux Network

If you have a subscription to Oracle Unbreakable Linux support, you can use the comprehensive resources of the Unbreakable Linux Network (ULN). ULN offers software patches, updates, and fixes for Oracle Linux and Oracle VM, as well as information on `yum`, Ksplice, and support policies. You can also download useful packages that are not included in the original distribution. The ULN Alert Notification Tool periodically checks with ULN and alerts you when updates are available. You can access ULN at https://linux.oracle.com/, where you will also find instructions for registering with ULN, for creating local `yum` repositories, and for switching from the Red Hat Network (RHN) to ULN.

If you want to use `yum` with ULN to manage your systems, you must register the systems with ULN and subscribe each system to one or more ULN channels. When you register a system with ULN, the channel that contains the latest version is chosen automatically according to the architecture and operating system revision of the system.

When you run `yum`, it connects to the ULN server repository and downloads the latest software packages in RPM format onto your system. `yum` then presents you with a list of the available packages so that you can choose which ones you want to install.

1.2 About ULN Channels

ULN provides more than 100 unique channels, which support the i386, x86_64, and ia64 architectures, for releases of Oracle Linux 4 update 6 and later.

You can choose for your system to remain at a specific OS revision, or you can allow the system to be updated with packages from later revisions.

You should subscribe to the channel that corresponds to the architecture of your system and the update level at which you want to maintain it. Patches and errata are available for specific revisions of Oracle Linux, but you do not need to upgrade from a given revision level to install these fixes. ULN channels also exist for MySQL, Oracle VM, OCFS2, RDS, and productivity applications.

The following table describes the main channels that are available.

Channel	Description
_latest	Provides all the packages in a distribution, including any errata that are also provided in the patch channel. Unless you explicitly specify the version, any package that you download on this channel will be the most recent that is available. If no vulnerabilities have been found in a package, the package version might be the same as that included in the original distribution. For other packages, the version will be the same as that provided in the patch channel for the highest update level. For example, the `ol6_arch_latest channel` for Oracle Linux 6 Update 3 contains a combination of the `ol6_u3_arch_base` and `ol6_u3_arch_patch` channels.
_base	Provides the packages for each major version and minor update of Oracle Linux and Oracle VM. This channel corresponds to the released ISO media image. For example, there is a base channel for each of the updates to Oracle Linux 6 as well as for Oracle Linux 6. Oracle does not publish security errata and bugfixes on these channels.
_patch	Provides only those packages that have changed since the initial release of a major or minor version of Oracle Linux or Oracle VM. The patch channel always provides the most recent version of a package, including all fixes that have been provided since the initial version was released.
_addons	Provides packages that are not included in the base distribution, such as the package that you can use to create a yum repository on Oracle Linux 6 or Oracle Linux 7.
_oracle	Provides freely downloadable RPMs from Oracle that you can install on Oracle Linux, such as ASMLib and Oracle Instant Client.
_optional	Provides optional packages for Oracle Linux 7 that have been sourced from upstream. This channel includes most development packages (`*-devel`).

Other channels may also be available, such as `_beta` channels for the beta versions of packages.

As each new major version or minor update of Oracle Linux becomes available, Oracle creates new base and patch channels for each supported architecture to distribute the new packages. The existing base and patch channels for the previous versions or updates remain available and do not include the new packages. The `_latest` channel distributes the highest possible version of any package, and tracks the top of the development tree independently of the update level.

> ⚠ **Caution**
>
> You can choose to maintain your system at a specific update level of Oracle Linux and selectively apply errata to that level by subscribing the system to the `_base` and `_patch` channels and unsubscribing it from the `_latest` channel. However, for Oracle Linux 7, patches are not added to the `_patch` channel for previous updates after a new update has been released. For example, after the release of Oracle Linux 7 Update 1, no further errata will be released on the `ol7_x86_64_u0_patch` channel.

> Oracle recommends that you keep you system subscribed to the `_latest` channel. If you unsubscribe from the `_latest` channel, your system will become vulnerable to security-related issues when a new update is released.

1.3 About Software Errata

Oracle releases important changes to Oracle Linux and Oracle VM software as individual package updates known as errata, which are made available for download on ULN before they are gathered into a release or are distributed via the `_patch` channel.

Errata packages can contain:

- Security advisories, which have names prefixed by `ELSA-*` (for Oracle Linux) and `OVMSA-*` (for Oracle VM).

- Bug fix advisories, which have names prefixed by `ELBA-*` and `OVMBA-*`.

- Feature enhancement advisories, which have names prefixed by `ELEA-*` and `OVMEA-*`.

To be notified when new errata packages are released, you can subscribe to the Oracle Linux and Oracle VM errata mailing lists at https://oss.oracle.com/mailman/listinfo/el-errata and https://oss.oracle.com/mailman/listinfo/oraclevm-errata.

If you are logged into ULN, you can also subscribe to these mailing lists by following the **Subscribe to Enterprise Linux Errata mailing list** and **Subscribe to Oracle VM Errata mailing list** links that are provided on the Errata tab.

1.4 Registering as a ULN User

When you register a system with ULN, your Oracle Single Signon (SSO) user name is also registered as your ULN user name. If you want to use ULN without first registering a system, you can register as a ULN user provided that you have a valid customer support identifier (CSI) for Oracle Linux support or Oracle VM support. To purchase Oracle Linux or Oracle VM support, go to the online Oracle Linux Store or contact your sales representative.

To register as a ULN user:

1. In a browser, go to https://linux.oracle.com/register.

2. If you do not have an SSO account, click **Create New Single Signon Account** and follow the onscreen instructions to create one.

   If you already have an SSO account, click **Sign On**.

3. Log in using your SSO user name and password.

4. On the Create New ULN User page, enter your CSI and click **Create New User**.

   > **Note**
   >
   > If no administrator is currently assigned to manage the CSI, you are prompted to click **Confirm** to become the CSI administrator. If you click **Cancel**, you cannot access the CSI administration feature. See Section 1.14, "About CSI Administration".
   >
   > If your user name already exists on the system, you are prompted to proceed to ULN by clicking the link **Unbreakable Linux Network**. If you enter a different

> CSI from your existing CSIs, your user name is associated with the new CSI in addition to your existing CSIs.

1.5 Registering an Oracle Linux 6 or Oracle Linux 7 System

To register an Oracle Linux 6 or Oracle Linux 7 system with ULN.

1. Run the `uln_register` command.

   ```
   # uln_register
   ```

   Alternatively, if you use the GNOME graphical user desktop, select **System > Administration > ULN Registration** on Oracle Linux 6 or **Applications > System Tools > ULN Registration** on Oracle Linux 7. You can also register your system with ULN if you configure networking when installing Oracle Linux 6 or Oracle Linux 7.

2. When prompted, enter your ULN user name, password, and customer support identifier (CSI).

3. Enter a name for the system that will allow you to identify it on ULN, and choose whether to upload hardware and software profile data that allows ULN to select the appropriate packages for the system.

4. If you have an Oracle Linux Premier Support account, you can choose to configure an Oracle Linux 6 or Oracle Linux 7 system that is running a supported kernel to receive kernel updates from Oracle Ksplice. See Section 3.2, "Registering to Use Ksplice Uptrack".

The `yum-rhn-plugin` is enabled and your system is subscribed to the appropriate software channels.

If you use a proxy server for Internet access, see Section 2.2.1, "Configuring Use of a Proxy Server".

1.6 Registering an Oracle Linux 4 or Oracle Linux 5 System

To register an Oracle Linux 4 or Oracle Linux 5 system with ULN.

1. Import the RPM GPG key.

   ```
   # rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY
   ```

2. Run the text-mode version of the `up2date` command.

   ```
   # up2date-nox --register
   ```

3. When prompted, enter your ULN user name, password, and CSI.

4. Enter the name of the system that will be displayed on ULN, and choose whether to upload hardware and software profile data that will allow ULN to select the appropriate packages for your system.

1.7 Configuring an Oracle Linux 5 System to Use yum with ULN

If your Oracle Linux 5 system is registered with ULN, you can use `yum` instead of `up2date` to download and install packages. If you have installed a full update since Oracle Linux 5 Update 6 was released on January 20, 2010, your system should already be able to use `yum` with ULN.

To enable `yum` support:

1. Install `yum-rhn-plugin`.

   ```
   # up2date --install yum-rhn-plugin
   ```

2. If your organization uses a proxy server as an intermediary for Internet access, specify the `enableProxy` and `httpProxy` settings in `/etc/sysconfig/rhn/up2date` as shown in this example.

```
enableProxy=1
httpProxy=http://proxysvr.yourdom.com:3128
```

If the proxy server requires authentication, additionally specify the `enableProxyAuth`, `proxyUser`, and `proxyPassword` settings:

```
enableProxy=1
enableProxyAuth=1
httpProxy=http://proxysvr.yourdom.com:3128
proxyUser=yumacc
proxyPassword=clydenw
```

> **Caution**
>
> All `yum` users require read access to `/etc/sysconfig/rhn/up2date`. If this file must be world-readable, do not use a password that is the same as any user's login password, and especially not `root`'s password.

With the plugin installed, you can immediately start to use `yum` instead of `up2date`.

1.8 Disabling Package Updates

To disable package updates by ULN (for example, if you have deleted your system from ULN), edit the `/etc/yum/pluginconf.d/rhnplugin.conf` file, and change the value of `enabled` flag from 1 to 0 in the `[main]` section, for example:

```
[main]
enabled = 0
gpgcheck = 1
```

To disable updates for particular packages, add an `exclude` statement to the `[main]` section of the `/etc/yum.conf` file. For example, to exclude updates for `VirtualBox` and `kernel`:

```
exclude=VirtualBox* kernel*
```

> **Note**
>
> Excluding certain packages from being updated can cause dependency errors for other packages. Your machine might also become vulnerable to security-related issues if you do not install the latest updates.

1.9 Subscribing Your System to ULN Channels

If you have registered your system with ULN, you can subscribe the system to the channels that are available for the level of support associated with the CSI.

To subscribe your system to ULN channels:

1. Log in to http://linux.oracle.com with your ULN user name and password.

2. On the Systems tab, click the link named for the system in the list of registered machines.

3. On the System Details page, click **Manage Subscriptions**.

4. On the System Summary page, select channels from the list of available or subscribed channels and click the arrows to move the channels between the lists.

5. When you have finished selecting channels, click **Save Subscriptions**.

1.10 Browsing and Downloading Errata Packages

You can browse the advisories that are available on ULN, and download the errata RPMs for the supported combinations of the software release and the system architecture.

To browse the advisories and download errata RPMs:

1. Log in to http://linux.oracle.com with your ULN user name and password.

2. Select the Errata tab.

    The Errata page displays a table of the available errata for all releases that are available on ULN.

3. On the Errata page, you can perform the following actions on the displayed errata:

    *   To sort the table of available errata, click the title of the **Type**, **Severity**, **Advisory**, **Systems Affected**, or **Release Date** column. Click the title again to reverse the order of sorting.

        > **Note**
        >
        > The **Systems Affected** column shows how many of your systems are potentially affected by an advisory.

    *   To display or hide advisories of different types, select or deselect the **Bug**, **Enhancement**, and **Security** check boxes and click **Go**.

    *   To display only advisories for a certain release of Oracle Linux or Oracle VM, select that release from the **Release** drop-down list and click **Go**.

    *   To search within the table, enter a string in the **Search** field and click **Go**.

4. To see more detail about an advisory and to download the RPMs:

    a. Click the link for the advisory.

    b. On the Errata Detail page for an advisory, you can download the RPMs for the supported releases and system architectures. The **Superseded By Advisory** column displays a link to the most recent advisory (if any) that replaces the advisory you are browsing.

1.11 Downloading Available Errata for a System

You can download a comma-separated values (CSV) report file of the errata that are available for your system and you can download errata RPMs.

To download a CSV report or the errata RPMs:

1. Log in to http://linux.oracle.com with your ULN user name and password.

2. On the Systems tab, click the link named for the system in the list of registered machines.

    The System Details page lists the available errata for the system in the Available Errata table, which might be split over several pages.

3. To download the CSV report file, click the link **Download All Available Errata for this System**.

4. To see more detail about an advisory and download the RPMs:

a. Click the link for the advisory.

b. On the System Errata Detail page for an advisory, you can download the RPMs for the affected releases and system architectures.

1.12 Updating System Details

If you have registered your system with ULN, you can update the details that ULN records for the system.

To update the details for your system:

1. Log in to http://linux.oracle.com with your ULN user name and password.

2. On the Systems tab, click the link named for the system in the list of registered machines.

3. On the System Details page, click **Edit**.

4. On the Edit System Properties page, you can change the name associated with your system, register it as a local yum server for your site, or change the CSI with which it is registered.

> **Note**
>
> You cannot change the CSI of a system unless it is registered to your user name.

5. When you have finished making changes, click **Apply Changes**.

1.13 Deleting a System

To delete a system that is registered on ULN:

1. Log in to http://linux.oracle.com with your ULN user name and password.

2. On the Systems tab, click the link named for the system in the list of registered machines.

3. On the System Details page, click **Delete**.

> **Note**
>
> You cannot delete a system unless it is registered to your user name.

4. When prompted to confirm the deletion, click **OK**.

1.14 About CSI Administration

The CSI administration feature of ULN provides a unified view of all of your organization's CSIs and the systems that are registered with those CSIs. To be able to manage the registered systems, you must become an administrator for one or more of your organization's CSIs. To be able to view and change the details of any system that is not registered to your ULN user name, you must become an administrator for the CSI under which that system is registered.

If you are registered as a CSI administrator, you can access the CSI Administration tab while logged in to ULN and perform the following tasks:

• Assign yourself as administrator of a CSI, or assign someone else as administrator of a CSI. See Section 1.14.1, "Becoming a CSI Administrator".

- List active CSIs, list the servers that are currently registered with an active CSI, and transfer those servers to another user or to another CSI. See Section 1.14.2, "Listing Active CSIs and Transferring Their Registered Servers".

- List expired CSIs, list the servers that are currently registered with an expired CSI, and transfer those servers to another user or to another CSI. See Section 1.14.3, "Listing Expired CSIs and Transferring Their Registered Servers".

- Remove yourself or someone else as administrator of a CSI. See Section 1.14.4, "Removing a CSI Administrator".

1.14.1 Becoming a CSI Administrator

You can become an administrator of a CSI in one of the following ways:

- When you register with ULN, if no administrator is currently assigned to manage the CSI, you are prompted to click **Confirm** to become the CSI administrator. If you click **Cancel**, you cannot access the CSI administration feature.

- When logged into ULN, if you access the System tab and no administrator is currently assigned to manage one of the CSIs for which you are registered, you are prompted to choose whether to become the CSI administrator.

  To become a CSI administrator:

  1. Click the red link labeled **enter the CSI you would like to be the administrator for in this page**.

  2. On the Add CSI page, verify the CSI and click **Confirm**.

     Note

     On the Systems page, the CSIs of all systems that have no assigned administrator are also shown in red.

- If you are already an administrator of a CSI, you can add yourself as administrator of another CSI provided that you have registered either a server or your ULN user name with the other CSI.

  To assign yourself as administrator of an additional CSI:

  1. Log in to ULN and select the CSI Administration tab.

  2. On the Managed CSIs page, click **Add CSI**.

  3. On the Assign Administrator page, enter the CSI, and click **Add**.

  4. If there are existing administrators, the page lists these administrators and prompts you to click **Confirm** to confirm your request. Each administrator is sent an email to inform them that you have added yourself as an administrator of the CSI.

- An administrator for a CSI can add you as an administrator for the same CSI.

  To assign another administrator to a CSI:

  1. Log in to ULN as administrator of the CSI, and select the CSI Administration tab.

  2. On the Managed CSIs page, click **List Administrators**.

  3. On the CSI Administrators page, click **Assign Administrator**.

4. On the Assign Administrator page in the Select New Administrator list, click the **+** icon that is next to the user name of the user that you want to add as an administrator. Their user name is added to the **Administrator** box.

5. If you administer more than one CSI, select the CSI that the user will administer from the **CSI** drop down list.

6. Click **Assign Administrator**.

> **Note**
>
> If you want to become the administrator of a CSI but the person to whom it is registered is no longer with your organization, contact an Oracle support representative to request that you be made the administrator for the CSI.

1.14.2 Listing Active CSIs and Transferring Their Registered Servers

To list details of the active CSIs for which you are the administrator:

1. Log in to ULN as administrator of the CSI, and select the CSI Administration tab.

2. On the Managed CSIs page in the Select Managed CSI Services pane, select the **Active** link. The Managed Active CSI Services pane displays the service details for each active CSI that you administer.

3. Click the **View # Server(s)** link to display the details of the servers that are registered to an active CSI.

4. On the Registered Servers page, you can transfer one or more systems to another user or to another CSI that you administer.

> **Note**
>
> If you transfer a system to another user, at least one of the following conditions must be true:
>
> • His or her user name must be registered to this CSI.
>
> • One or more of the servers, for which they are the owner, must be registered to this CSI.
>
> • He or she must be an administrator of at least one CSI for which you are also an administrator.

To transfer systems to another user:

a. Select the **Transfer System** check boxes for the systems that you want to transfer.

b. Click **Transfer Selected Systems to Another Owner**.

c. On the Transfer Registered System(s) - Owner page in the Transfer To column, click the red arrow icon that is next to the user name of the user to whom you want to transfer ownership.

d. On the Confirm Transfer Profile - Owner page, click **Apply Changes** to confirm the transfer to the new owner.

To transfer systems to another CSI:

a. Select the **Transfer System** check boxes for the systems that you want to transfer.

b. Click **Transfer Selected Systems to Another CSI**.

c. On the Transfer Registered System(s) - CSI page in the Transfer To column, click the red arrow icon that is next to the CSI to which you want to transfer the systems.

d. On the Confirm Transfer Profile - CSI page, click **Apply Changes** to confirm the transfer to the new CSI.

1.14.3 Listing Expired CSIs and Transferring Their Registered Servers

To list details of the expired CSIs for which you are the administrator:

1. Log in to ULN as administrator of the CSI, and select the CSI Administration tab.

2. On the Managed CSIs page in the Select Managed CSI Services pane, select the **Expired** link. The Managed Expired CSI Services pane displays the service details for each expired CSI that you administer.

3. Click the **View # Server(s)** link to display the details of the servers that are registered to an expired CSI.

4. On the Registered Servers page, you can transfer one or more systems to another user or to another CSI that you administer.

> **Note**
>
> If you transfer a system to another user, at least one of the following conditions must be true:
>
> - His or her user name must be registered to this CSI.
>
> - One or more of the servers, for which they are the owner, must be registered to this CSI.
>
> - He or she must be an administrator of at least one CSI for which you are also an administrator.

To transfer systems to another user:

a. Select the **Transfer System** check boxes for the systems that you want to transfer.

b. Click **Transfer Selected Systems to Another Owner**.

c. On the Transfer Registered System(s) - Owner page in the Transfer To column, click the red arrow icon that is next to the user name of the user to whom you want to transfer ownership.

d. On the Confirm Transfer Profile - Owner page, click **Apply Changes** to confirm the transfer to the new owner.

To transfer systems to another CSI:

a. Select the **Transfer System** check boxes for the systems that you want to transfer.

b. Click **Transfer Selected Systems to Another CSI**.

c. On the Transfer Registered System(s) - CSI page in the Transfer To column, click the red arrow icon that is next to the CSI to which you want to transfer the systems.

    d. On the Confirm Transfer Profile - CSI page, click **Apply Changes** to confirm the transfer to the new CSI.

1.14.4 Removing a CSI Administrator

To remove an administrator who is registered for a CSI:

1. Log in to ULN and select the CSI Administration tab.

2. On the Managed CSIs page, click **List Administrators**.

3. On the CSI Administrators page in the Delete? column, click the trash can icon that is next to the user name of the user that you want to remove as administrator for the CSI specified in the same row.

4. When prompted to confirm that you want to revoke administration privileges for the CSI from that user, click **OK**.

1.15 Switching from RHN to ULN

> **Note**
>
> This procedure is for a Red Hat Enterprise Linux 6 system. For details of equivalent procedures for Red Hat Enterprise Linux 3, 4, and 5, see http://linux.oracle.com/switch.html.
>
> If you have an Oracle Linux 6 or Oracle Linux 7 system that is registered with the Red Hat Network (RHN), you can use the `uln_register` utility to register it as described in Section 1.5, "Registering an Oracle Linux 6 or Oracle Linux 7 System".
>
> You must have a ULN account before you can register a system with ULN. You can create a ULN account at http://linux.oracle.com/register.

To register your system with ULN instead of RHN:

1. Download the `uln_register.tgz` package from http://linux-update.oracle.com/rpms to a temporary directory.

   If the `rhn-setup-gnome` package is already installed on your system, also download the `uln_register-gnome.tgz` from the same URL.

2. Extract the packages using the following command.

   ```
   # tar -xzf uln_register.tgz
   ```

   If the `rhn-setup-gnome` package is installed on your system, extract the packages from `uln_register-gnome.tgz`.

   ```
   # tar -xzf uln_register-gnome.tgz
   ```

3. Change to the `uln_migrate` directory and install the registration packages.

   ```
   # cd ./uln_migrate
   # rpm -Uvh *.rpm
   ```

4. Run the `uln_register` command.

   ```
   # uln_register
   ```

5. Follow the instructions on the screen to complete the registration. The `uln_register` utility collects information about your system and uploads it to Oracle.

1.16 For More Information About ULN

You can find out more information about ULN at https://linux.oracle.com/.

Chapter 2 Yum

Table of Contents

This chapter describes how you can use the `yum` utility to install and upgrade software packages.

2.1 About Yum

Oracle Linux provides the `yum` utility which you can use to install or upgrade RPM packages. The main benefit of using yum is that it also installs or upgrades any package dependencies. `yum` downloads the packages from repositories such as those that are available on the Oracle Linux Yum Server, but you can also set up your own repositories on systems that do not have Internet access.

The Oracle Linux Yum Server is a convenient way to install Oracle Linux and Oracle VM packages, including bug fixes, security fixes and enhancements, rather than installing them from installation media. You can access the server at http://yum.oracle.com/.

You can also subscribe to the Oracle Linux and Oracle VM errata mailing lists to be notified when new packages are released. You can access the mailing lists at https://oss.oracle.com/mailman/listinfo/el-errata and https://oss.oracle.com/mailman/listinfo/oraclevm-errata.

If you have registered your system with the Unbreakable Linux Network (ULN), you can use `yum` with ULN channels to maintain the software on your system, as described in Chapter 1, *The Unbreakable Linux Network*.

2.2 Yum Configuration

The main configuration file for yum is `/etc/yum.conf`. The global definitions for yum are located under the `[main]` section heading of the yum configuration file. The following table lists the important directives.

Directive	Description
`cachedir`	Directory used to store downloaded packages.
`debuglevel`	Logging level, from 0 (none) to 10 (all).
`exactarch`	If set to 1, only update packages for the correct architecture.
`exclude`	A space separated list of packages to exclude from installs or updates, for example: `exclude=VirtualBox-4.? kernel*`.

Directive	Description
gpgcheck	If set to 1, verify the authenticity of the packages by checking the GPG signatures. You might need to set gpgcheck to 0 if a package is unsigned, but you should be wary that the package could have been maliciously altered.
gpgkey	Pathname of the GPG public key file.
installonly_limit	Maximum number of versions that can be installed of any one package.
keepcache	If set to 0, remove packages after installation.
logfile	Pathname of the yum log file.
obsoletes	If set to 1, replace obsolete packages during upgrades.
plugins	If set to 1, enable plugins that extend the functionality of yum.
proxy	URL of a proxy server including the port number. See Section 2.2.1, "Configuring Use of a Proxy Server".
proxy_password	Password for authentication with a proxy server.
proxy_username	User name for authentication with a proxy server.
reposdir	Directories where yum should look for repository files with a .repo extension. The default directory is /etc/yum.repos.d.

See the yum.conf(5) manual page for more information.

The following listing shows an example [main] section from the yum configuration file.

```
[main]
cachedir=/var/cache/yum
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgkey=file://media/RPM-GPG-KEY
gpgcheck=1
pligins=1
installonly_limit=3
```

It is possible to define repositories below the [main] section in /etc/yum.conf or in separate repository configuration files. By default, yum expects any repository configuration files to be located in the /etc/yum.repos.d directory unless you use the reposdir directive to define alternate directories.

2.2.1 Configuring Use of a Proxy Server

If your organization uses a proxy server as an intermediary for Internet access, specify the proxy setting in /etc/yum.conf as shown in the following example.

```
proxy=http://proxysvr.yourdom.com:3128
```

If the proxy server requires authentication, additionally specify the proxy_username, and proxy_password settings.

```
proxy=http://proxysvr.yourdom.com:3128
proxy_username=yumacc
proxy_password=clydenw
```

If you use the yum plugin (yum-rhn-plugin) to access the ULN, specify the enableProxy and httpProxy settings in /etc/sysconfig/rhn/up2date as shown in this example.

```
enableProxy=1
httpProxy=http://proxysvr.yourdom.com:3128
```

If the proxy server requires authentication, additionally specify the `enableProxyAuth`, `proxyUser`, and `proxyPassword` settings.

```
enableProxy=1
httpProxy=http://proxysvr.yourdom.com:3128
enableProxyAuth=1
proxyUser=yumacc
proxyPassword=clydenw
```

> **⚠ Caution**
>
> All `yum` users require read access to `/etc/yum.conf` or `/etc/sysconfig/rhn/up2date`. If these files must be world-readable, do not use a proxy password that is the same as any user's login password, and especially not `root`'s password.

2.2.2 Yum Repository Configuration

The yum configuration file or yum repository configuration files can contain one or more sections that define repositories.

The following table lists the basic directives for a repository.

Directive	Description
`baseurl`	Location of the repository channel (expressed as a `file://`, `ftp://`, `http://`, or `https://` address). This directive must be specified.
`enabled`	If set to 1, permit `yum` to use the channel.
`name`	Descriptive name for the repository channel. This directive must be specified.

Any other directive that appears in this section overrides the corresponding global definition in `[main]` section of the yum configuration file. See the `yum.conf(5)` manual page for more information.

The following listing shows an example repository section from a configuration file.

```
[ol6_u2_base]
name=Oracle Linux 6 U2 - $basearch - base
baseurl=http://yum.oracle.com/repo/OracleLinux/OL6/2/base/$basearch
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=1
```

In this example, the values of `gpgkey` and `gpgcheck` override any global setting. `yum` substitutes the name of the current system's architecture for the variable `$basearch`.

2.3 Downloading the Oracle Linux Yum Server Repository Files

> **Note**
>
> The following procedure assumes that `yum` on your system is configured to expect to find repository files in the default `/etc/yum.repos.d` directory.

To download the Oracle Linux Yum Server repository configuration file:

1. As `root`, change directory to `/etc/yum.repos.d`.

```
# cd /etc/yum.repos.d
```

2. Use the `wget` utility to download the repository configuration file that is appropriate for your system.

```
# wget http://yum.oracle.com/public-yum-release.repo
```

For Oracle Linux 7, enter:

```
# wget http://yum.oracle.com/public-yum-ol7.repo
```

The `/etc/yum.repos.d` directory is updated with the repository configuration file, in this example, `public-yum-ol7.repo`.

3. You can enable or disable repositories in the file by setting the value of the `enabled` directive to 1 or 0 as required.

2.4 Using Yum from the Command Line

The following table shows some examples of common tasks that you can perform using `yum`.

Command	Description
`yum repolist`	Lists all enabled repositories.
`yum list`	Lists all packages that are available in all enabled repositories and all packages that are installed on your system.
`yum list installed`	Lists all packages that are installed on your system.
`yum list available`	Lists all packages that are available to be installed in all enabled repositories.
`yum search string`	Searches the package descriptions for the specified string.
`yum provides feature`	Finds the name of the package to which the specified file or feature belongs. For example: `yum provides /etc/sysconfig/atd`
`yum info package`	Displays detailed information about a package. For example: `yum info bind`
`yum install package`	Installs the specified package, including packages on which it depends. For example: `yum install ocfs2-tools`
`yum check-update`	Checks whether updates exist for packages that are already installed on your system.
`yum update package`	Updates the specified package, including packages on which it depends. For example: `yum upgrade nfs-utils`
`yum update`	Updates all packages, including packages on which they depend.
`yum remove package`	Removes the specified package. For example: `yum erase nfs-utils`
`yum erase package`	Removes the specified package. This command has the same effect as the `yum remove` command.

Command	Description
`yum update`	Updates all packages, including packages on which they depend.
`yum clean all`	Removes all cached package downloads and cached headers that contain information about remote packages. Running this command can help to clear problems that can result from unfinished transactions or out-of-date headers.
`yum help`	Displays help about `yum` usage.
`yum help command`	Displays help about the specified `yum` command. For example: `yum help upgrade`
`yum shell`	Runs the `yum` interactive shell.

See the `yum(8)` manual page for more information.

To list the files in a package, use the `repoquery` utility, which is included in the `yum-utils` package. For example, the following command lists the files that the `btrfs-progs` package provides.

```
# repoquery -l btrfs-progs
/sbin/btrfs
/sbin/btrfs-convert
/sbin/btrfs-debug-tree
.
.
.
```

Note

`yum` makes no distinction between installing and upgrading a kernel package. `yum` always installs a new kernel regardless of whether you specify `update` or `install`.

2.5 Yum Groups

A set of packages can themselves be organized as a *yum group*. Examples include the groups for Eclipse, fonts, and system administration tools. The following table shows the `yum` commands that you can use to manage these groups.

Command	Description
`yum grouplist`	Lists installed groups and groups that are available for installation.
`yum groupinfo groupname`	Displays detailed information about a group.
`yum groupinstall groupname`	Installs all the packages in a group.
`yum groupupdate groupname`	Updates all the packages in a group.
`yum groupremove groupname`	Removes all the packages in a group.

2.6 Using the Yum Security Plugin

The security plugin is integrated with `yum` in Oracle Linux 7 and allows you to obtain a list of all of the errata that are available for your system, including security updates. You can also use Oracle Enterprise Manager 12c Cloud Control or management tools such as Katello, Pulp, Red Hat Satellite, Spacewalk, and SUSE Manager to extract and display information about errata.

To list the errata that are available for your system, enter:

```
# yum updateinfo list
ELBA-2012-1518 bugfix          NetworkManager-1:0.8.1-34.el6_3.x86_64
ELBA-2012-1518 bugfix          NetworkManager-glib-1:0.8.1-34.el6_3.x86_64
ELBA-2012-1518 bugfix          NetworkManager-gnome-1:0.8.1-34.el6_3.x86_64
ELBA-2012-1457 bugfix          ORBit2-2.14.17-3.2.el6_3.x86_64
ELBA-2012-1457 bugfix          ORBit2-devel-2.14.17-3.2.el6_3.x86_64
ELSA-2013-0215 Important/Sec. abrt-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-addon-ccpp-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-addon-kerneloops-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-addon-python-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-cli-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-desktop-2.0.8-6.0.1.el6_3.2.x86_64
...
```

The output from the command sorts the available errata in order of their IDs, and it also specifies whether each erratum is a security patch (*severity*/Sec.), a bug fix (bugfix), or a feature enhancement (enhancement). Security patches are listed by their severity: Important, Moderate, or Low.

You can use the --sec-severity option to filter the security errata by severity, for example:

```
# yum updateinfo list --sec-severity=Moderate
ELSA-2013-0269 Moderate/Sec. axis-1.2.1-7.3.el6_3.noarch
ELSA-2013-0668 Moderate/Sec. boost-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-date-time-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-devel-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-filesystem-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-graph-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-iostreams-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-program-options-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-python-1.41.0-15.el6_4.x86_64
...
```

To list the security errata by their Common Vulnerabilities and Exposures (CVE) IDs instead of their errata IDs, specify the keyword cves as an argument:

```
# yum updateinfo list cves
 CVE-2012-5659 Important/Sec. abrt-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5660 Important/Sec. abrt-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5659 Important/Sec. abrt-addon-ccpp-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5660 Important/Sec. abrt-addon-ccpp-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5659 Important/Sec. abrt-addon-kerneloops-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5660 Important/Sec. abrt-addon-kerneloops-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5659 Important/Sec. abrt-addon-python-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5660 Important/Sec. abrt-addon-python-2.0.8-6.0.1.el6_3.2.x86_64
...
```

Similarly, the keywords bugfix, enhancement, and security filter the list for all bug fixes, enhancements, and security errata.

You can use the --cve option to display the errata that correspond to a specified CVE, for example:

```
# yum updateinfo list --cve CVE-2012-2677
ELSA-2013-0668 Moderate/Sec. boost-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-date-time-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-devel-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-filesystem-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-graph-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-iostreams-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-program-options-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-python-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-regex-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-serialization-1.41.0-15.el6_4.x86_64
```

```
ELSA-2013-0668 Moderate/Sec. boost-signals-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-system-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-test-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-thread-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-wave-1.41.0-15.el6_4.x86_64
updateinfo list done
```

To display more information, specify `info` instead of `list`, for example:

```
# yum updateinfo info --cve CVE-2012-2677
===============================================================================
   boost security update
===============================================================================
  Update ID : ELSA-2013-0668
    Release : Oracle Linux 6
       Type : security
     Status : final
     Issued : 2013-03-21
       CVEs : CVE-2012-2677
Description : [1.41.0-15]
            : - Add in explicit dependences between some boost
            :   subpackages
            :
            : [1.41.0-14]
            : - Build with -fno-strict-aliasing
            :
            : [1.41.0-13]
            : - In Boost.Pool, be careful not to overflow
            :   allocated chunk size (boost-1.41.0-pool.patch)
            :
            : [1.41.0-12]
            : - Add an upstream patch that fixes computation of
            :   CRC in zlib streams.
            : - Resolves: #707624
   Severity : Moderate
updateinfo info done
```

To update all packages for which security-related errata are available to the latest versions of the packages, even if those packages include bug fixes or new features but not security errata, enter:

```
# yum --security update
```

To update all packages to the latest versions that contain security errata, ignoring any newer packages that do not contain security errata, enter:

```
# yum --security update-minimal
```

To update all kernel packages to the latest versions that contain security errata, enter:

```
# yum --security update-minimal kernel*
```

You can also update only those packages that correspond to a CVE or erratum, for example:

```
# yum update --cve CVE-2012-3954
```

```
# yum update --advisory ELSA-2012-1141
```

> **Note**
>
> Some updates might require you to reboot the system. By default, the boot manager will automatically enable the most recent kernel version.

For more information, see the `yum-security(8)` manual page.

2.7 Switching CentOS or Scientific Linux Systems to Use the Oracle Linux Yum Server

You can use the `centos2ol.sh` script to convert CentOS 5 and 6 or Scientific Linux 5 and 6 systems to Oracle Linux. The script configures `yum` to use the Oracle Linux Yum Server and installs a few additional packages that are required. There is no need to reboot the system.

To perform the switch to Oracle Linux, run the following commands as `root`:

```
# curl -O https://linux.oracle.com/switch/centos2ol.sh
# sh centos2ol.sh
```

For more information, see https://linux.oracle.com/switch/centos/.

2.8 Creating and Using a Local ULN Mirror

For information on how to create and use a yum server that acts as a local mirror of the ULN channels, see Creating and Using a Local ULN Mirror in the *Oracle Linux Unbreakable Linux Network User's Guide*.

2.9 Creating a Local Yum Repository Using an ISO Image

> **Note**
>
> The system must have sufficient storage space to host a full Oracle Linux Media Pack DVD image (approximately 3.5 GB for Oracle Linux Release 6 Update 3).

To create a local yum repository (for example, if a system does not have Internet access):

1. On a system with Internet access, download a full Oracle Linux DVD image from the Oracle Software Delivery Cloud at http://edelivery.oracle.com/linux onto removable storage (such as a USB memory stick). For example, `V33411-01.iso` contains the Oracle Linux Release 6 Update 3 Media Pack for x86 (64 bit).

   > **Note**
   >
   > You can verify that the ISO was copied correctly by comparing its checksum with the digest value that is listed on `edelivery.oracle.com`, for example:
   >
   > ```
   > # sha1sum V33411-01.iso
   > 7daae91cc0437f6a98a4359ad9706d678a9f19de V33411-01.iso
   > ```

2. Transfer the removable storage to the system on which you want to create a local yum repository, and copy the DVD image to a directory in a local file system.

   ```
   # cp /media/USB_stick/V33411-01.iso /ISOs
   ```

3. Create a suitable mount point, for example `/var/OSimage/OL6.3_x86_64`, and mount the DVD image on it.

   ```
   # mkdir -p /var/OSimage/OL6.3_x86_64
   # mount -o loop,ro /ISOs/V33411-01.iso /var/OSimage/OL6.3_x86_64
   ```

   > **Note**
   >
   > Include the read-only mount option (`ro`) to avoid changing the contents of the ISO by mistake.

4. Create an entry in `/etc/fstab` so that the system always mounts the DVD image after a reboot.

```
/ISOs/V33411-01.iso /var/OSimage/OL6.3_x86_64 iso9660 loop,ro 0 0
```

5. In the `/etc/yum.repos.d` directory, edit the existing repository files, such as `public-yum-ol6.repo` or `ULN-base.repo`, and disable all entries by setting `enabled=0`.

6. Create the following entries in a new repository file (for example, `/etc/yum.repos.d/OL63.repo`).

```
[OL63]
name=Oracle Linux 6.3 x86_64
baseurl=file:///var/OSimage/OL6.3_x86_64
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=1
```

7. Clean up the `yum` cache.

```
# yum clean all
```

8. Test that you can use `yum` to access the repository.

```
# yum repolist
Loaded plugins: refresh-packagekit, security
...
repo id                          repo name                              status
OL63                             Oracle Linux 6.3 x86_64                25,459
repolist: 25,459
```

2.10 Setting up a Local Yum Server Using an ISO Image

To set up a local yum server (for example, if you have a network of systems that do not have Internet access):

1. Choose one of the systems to be the yum server, and create a local yum repository on it as described in Section 2.9, "Creating a Local Yum Repository Using an ISO Image".

2. Install the Apache HTTP server from the local yum repository.

```
# yum install httpd
```

3. If SELinux is enabled in enforcing mode on your system:

   a. Use the `semanage` command to define the default file type of the repository root directory hierarchy as `httpd_sys_content_t`:

   ```
   # /usr/sbin/semanage fcontext -a -t httpd_sys_content_t "/var/OSimage(/.*)?"
   ```

   b. Use the `restorecon` command to apply the file type to the entire repository.

   ```
   # /sbin/restorecon -R -v /var/OSimage
   ```

   > **Note**
   >
   > The `semanage` and `restorecon` commands are provided by the `policycoreutils-python` and `policycoreutils` packages.

4. Create a symbolic link in `/var/www/html` that points to the repository:

```
# ln -s /var/OSimage /var/www/html/OSimage
```

5. Edit the HTTP server configuration file, `/etc/httpd/conf/httpd.conf`, as follows:

a. Specify the resolvable domain name of the server in the argument to `ServerName`.

```
ServerName server_addr:80
```

If the server does not have a resolvable domain name, enter its IP address instead.

b. Verify that the setting of the `Options` directive in the `<Directory "/var/www/html">` section specifies `Indexes` and `FollowSymLinks` to allow you to browse the directory hierarchy, for example:

```
Options Indexes FollowSymLinks
```

c. Save your changes to the file.

6. Start the Apache HTTP server, and configure it to start after a reboot.

```
# systemctl start httpd
# systemctl enable httpd
```

7. If you have enabled a firewall on your system, configure it to allow incoming HTTP connection requests on TCP port 80, for example:

```
# firewall-cmd --zone=zone --add-port=80/tcp
# firewall-cmd --permanent --zone=zone --add-port=80/tcp
```

8. Edit the repository file on the server (for example, `/etc/yum.repos.d/OL63.repo`):

```
[OL63]
name=Oracle Linux 6.3 x86_64
baseurl=http://server_addr/OSimage/OL6.3_x86_64
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=1
```

Replace `server_addr` with the IP address or resolvable host name of the local yum server.

9. On each client, copy the repository file from the server to the `/etc/yum.repos.d` directory.

10. In the `/etc/yum.repos.d` directory, edit any other repository files, such as `public-yum-ol6.repo` or `ULN-base.repo`, and disable all entries by setting `enabled=0`.

11. On the server and each client, test that you can use `yum` to access the repository.

```
# yum repolist
Loaded plugins: refresh-packagekit, security
...
repo id                         repo name                               status
OL63                            Oracle Linux 6.3 x86_64                 25,459
repolist: 25,459
```

2.11 For More Information About Yum

For more information about yum, see http://yum.baseurl.org/.

For more information about how to download the latest packages from the Unbreakable Linux Network and make the packages available through a local yum server, see http://www.oracle.com/technetwork/articles/servers-storage-admin/yum-repo-setup-1659167.html.

Chapter 3 Ksplice Uptrack

Table of Contents

This chapter describes how to configure Ksplice Uptrack to update the kernel on a running system.

> **Note**
>
> An enhanced version of the Ksplice client is available that can patch shared
> libraries for user-space processes that are running on an Oracle Linux 7 system.
> For more information, see About the Enhanced Ksplice Client in the *Oracle Linux
> Ksplice User's Guide*.

3.1 About Ksplice Uptrack

Ksplice Uptrack can update a running Linux kernel without requiring an immediate reboot of the system.
You can apply Ksplice updates to both the Unbreakable Enterprise Kernel and the Red Hat Compatible
Kernel. Oracle creates each Ksplice patch from a kernel update that originates from either Oracle or the
Linux kernel community. Ksplice Uptrack allows you to apply the latest kernel security errata for Common
Vulnerabilities and Exposures (CVEs) without halting the system or restarting applications. Ksplice Uptrack
applies the update patches in the background with a negligible impact, usually consisting of a pause of at
most a few milliseconds. Ksplice Uptrack allows you to keep your systems secure and highly available.
You can use Ksplice Uptrack and still upgrade your kernel using your usual mechanism, such as by using
`yum`.

3.1.1 Supported Kernels

You can use Ksplice Uptrack to bring the following Oracle Linux kernels up to date with the latest important
security and bug fix patches:

- All Oracle Unbreakable Enterprise Kernel versions for Oracle Linux 5, Oracle Linux 6, and Oracle Linux
  7 starting with 2.6.32-100.28.9 (released March 16, 2011).

- All Oracle Linux 6 and Oracle Linux 7 kernels starting with the official release.

- All Oracle Linux 5 Red Hat Compatible Kernels starting with Oracle Linux 5.4 (2.6.18-164.el5, released
  September 9, 2009).

- All Oracle Linux 5 Red Hat Compatible Kernels with bug fixes added by Oracle starting with Oracle Linux
  5.6 (2.6.18-238.0.0.0.1.el5, released January 22, 2011).

To confirm whether a particular kernel is supported, install the Uptrack client on a system that is running the kernel.

If you have a question about supported kernels, send e-mail to ksplice-support_ww@oracle.com.

3.2 Registering to Use Ksplice Uptrack

When you register your systems with ULN, you can opt to use Oracle Ksplice if you have an Oracle Linux Premier Support account. If you choose to use Ksplice, you can subscribe your systems to the Ksplice for Oracle Linux channel and install the Ksplice Uptrack software on them. To install the `uptrack` package after registration is complete, you can use `yum` on an Oracle Linux 6 or Oracle Linux 7 system or `up2date` on an Oracle Linux 5 system. The Uptrack client downloads the access key from ULN and automatically configures itself so that you can immediately begin to use Ksplice Uptrack.

If you already have an account on ULN, you can register your system to use Ksplice Uptrack at http://linux.oracle.com.

1. From your browser, log in to ULN with your existing user name and password. If your subscription grants you access to Ksplice, the ULN home page displays the **Ksplice Uptrack Registration** button.

2. Click **Ksplice Uptrack Registration**. The screen displays all valid Customer Support Identifiers (CSIs) for your account.

3. Select the CSI that you want to use and click Register. The screen displays an acknowledgment that a Ksplice account has been created and that an e-mail containing the Ksplice access key, a temporary password for Ksplice, and a URL for confirming your registration has been sent to your e-mail account.

4. When you receive the e-mail, open the URL that it contains.

5. Complete the form to confirm your registration, and click **Continue**.

After registering to use Ksplice Uptrack, you can log in at https://uptrack.ksplice.com using your e-mail address as your user name, and the temporary password. You must change your password when you first log in. You can view the status of your registered systems, the patches that have been applied, and the patches that are available. You can also create access control groups for your registered systems.

3.3 Installing Ksplice Uptrack

If you have an Oracle Linux Premier Support account and you have registered to use Oracle Ksplice, you can configure your registered systems to use Ksplice Uptrack through the Ksplice for Oracle Linux channel on ULN by using `yum`.

The system on which you want to install Ksplice Uptrack must meet the following criteria:

• The system must be registered with ULN.

• The operating system must be Oracle Linux 5, Oracle Linux 6, or Oracle Linux 7 with a supported version of either the Unbreakable Enterprise Kernel or the Red Hat Compatible Kernel installed. You can verify the kernel version by using the `uname -a` command. See Section 3.1.1, "Supported Kernels".

• The kernel that is running currently is assumed to be the one that you want to update. Ksplice Uptrack applies updates only to the running kernel.

• The system must have access to the Internet.

To install Ksplice Uptrack from ULN:

1. Log in as `root` on the system.

2. If you use an Internet proxy, configure the HTTP and HTTPS settings for the proxy in the shell.

   - For the `sh`, `ksh`, or `bash` shells, use commands such as the following:

   ```
   # http_proxy=http://proxy_URL:http_port
   # https_proxy=http://proxy_URL:https_port
   # export http_proxy https_proxy
   ```

   For the `csh` shell, use commands such as the following:

   ```
   # setenv http_proxy=http://proxy_URL:http_port
   # setenv https_proxy=http://proxy_URL:https_port
   ```

3. Using a browser, log in at http://linux.oracle.com with the ULN user name and password that you used to register the system, and perform the following steps:

   a. On the Systems tab, click the link named for your system in the list of registered machines.

   b. On the System Details page, click **Manage Subscriptions**.

   c. On the System Summary page, select the Ksplice for Oracle Linux channel for the correct release and your system's architecture (`i386` or `x86_64`) from the list of available channels and click the right arrow (**>**) to move it to the list of subscribed channels.

   d. Click **Save Subscriptions** and log out of the ULN.

4. On your system, use `yum` to install the `uptrack` package.

   ```
   # yum install -y uptrack
   ```

   The access key for Ksplice Uptrack is retrieved from ULN and added to `/etc/uptrack/uptrack.conf`, for example:

   ```
   [Auth]
   accesskey = 0e1859ad8aea14b0b4306349142ce9160353297daee30240dab4d61f4ea4e59b
   ```

5. To enable the automatic installation of updates, change the following entry in `/etc/uptrack/uptrack.conf`:

   ```
   autoinstall = no
   ```

   so that it reads:

   ```
   autoinstall = yes
   ```

For information about configuring Ksplice Uptrack, see Section 3.4, "Configuring Ksplice Uptrack".

For information about managing Ksplice updates, see Section 3.5, "Managing Ksplice Updates".

3.4 Configuring Ksplice Uptrack

The configuration file for Ksplice Uptrack is `/etc/uptrack/uptrack.conf`. You can modify this file to configure a proxy server, to install updates automatically at boot time, or to check for and apply new updates automatically.

Ksplice Uptrack communicates with the Uptrack server by connecting to `https://updates.ksplice.com:443`. You can either configure your firewall to allow connection via port 443,

or you can configure Ksplice Uptrack to use a proxy server. To configure Ksplice Uptrack to use a proxy server, set the following entry in `/etc/uptrack/uptrack.conf`:

```
https_proxy = https://proxy_URL:https_port
```

You receive e-mail notification when Ksplice updates are available for your system.

To make Ksplice Uptrack install all updates automatically as they become available, set the following entry:

```
autoinstall = yes
```

> **Note**
>
> Enabling automatic installation of updates does not automatically update Ksplice Uptrack itself. Oracle notifies you by e-mail when you can upgrade the Ksplice Uptrack software using `yum`.

To install updates automatically at boot time, the following entry must appear in `/etc/uptrack/uptrack.conf`:

```
install_on_reboot = yes
```

When you boot the system into the same kernel, the `/etc/init.d/uptrack` script reapplies the installed Ksplice updates to the kernel.

To prevent Ksplice Uptrack from automatically reapplying updates to the kernel when you reboot the system, set the entry to:

```
install_on_reboot = no
```

To install all available updates at boot time, even if you boot the system into a different kernel, uncomment the following entry in `/etc/uptrack/uptrack.conf`:

```
#upgrade_on_reboot = yes
```

so that it reads:

```
upgrade_on_reboot = yes
```

3.5 Managing Ksplice Updates

Ksplice patches are stored in `/var/cache/uptrack`. Following a reboot, Ksplice Uptrack automatically re-applies these patches very early in the boot process before the network is configured, so that the system is hardened before any remote connections can be established.

To list the available Ksplice updates, use the `uptrack-upgrade` command:

```
# uptrack-upgrade -n
```

To install all available Ksplice updates, enter:

```
# uptrack-upgrade -y
```

To install an individual Ksplice update, specify the update's ID as the argument (in this example, the ID is `dfvn0zq8`):

```
# uptrack-upgrade dfvn0zq8
```

After Ksplice has applied updates to a running kernel, the kernel has an effective version that is different from the original boot version displayed by the `uname -a` command. Use the `uptrack-uname` command to display the effective version of the kernel:

```
# uptrack-uname -a
```

`uptrack-uname` supports the commonly used `uname` flags, including `-a` and `-r`, and provides a way for applications to detect that the kernel has been patched. The effective version is based on the version number of the latest patch that Ksplice Uptrack has applied to the kernel.

To view the updates that Ksplice has made to the running kernel:

```
# uptrack-show
```

To view the updates that are available to be installed:

```
# uptrack-show --available
```

To remove all updates from the kernel:

```
# uptrack-remove --all
```

To prevent Ksplice Uptrack from reapplying the updates at the next system reboot, create the empty file `/etc/uptrack/disable`:

```
# touch /etc/uptrack/disable
```

Alternatively, specify `nouptrack` as a parameter on the boot command line when you next restart the system.

3.6 Patching and Updating Your System

Ksplice patches allow you to keep a system up to date while it is running. You should also use `yum` or `rpm` to install the regular kernel RPM packages for released errata that are available from the Unbreakable Linux Network (ULN) or the Oracle Linux Yum Server. Your system will then be ready for the next maintenance window or reboot. When you do restart the system, you can boot it from a newer kernel version. Ksplice Uptrack uses the new kernel as a baseline for applying patches as they become available.

3.7 Removing the Ksplice Uptrack software

To remove the Ksplice Uptrack software from a system, enter:

```
# yum -y remove uptrack
```

3.8 About Ksplice Offline Client

Ksplice Offline Client removes the requirement for a server on your intranet to have a direct connection to the Oracle Uptrack server. All available Ksplice updates for each supported kernel version are bundled into an RPM that is specific to that version, and this package is updated every time that a new Ksplice patch becomes available for the kernel.

A Ksplice offline client does not require a network connection to be able to apply the update package to the kernel. For example, you could use `rpm` to install the update package from a memory stick. However, a more usual arrangement would be to create a local yum server that acts as a mirror of the Ksplice for Oracle Linux channels on ULN. At regular intervals, you download the latest Ksplice update packages to this server. After installing Ksplice Offline Client on your local systems, they can connect to the local ULN mirror to receive updates. They do not require access the Oracle Uptrack server.

> **Note**
>
> You cannot use the web interface or the Ksplice Uptrack API to monitor systems that are running Ksplice Offline Client as such systems are not registered with https://uptrack.ksplice.com.

3.8.1 Modifying a Local Yum Server to Act as a Ksplice Mirror

The system that you want to set up as a Ksplice mirror must meet the following criteria:

- You must have registered the system with ULN.

- You must have configured the system as a local ULN mirror. See Creating and Using a Local ULN Mirror in the *Oracle Linux Unbreakable Linux Network User's Guide*.

- The system should also have enough disk space to store copies of the packages that it hosts. As a general rule, you require between 6 and 10 GB of space for the packages of each major release.

To set up a local yum server as a Ksplice mirror:

1. Using a browser, log in at http://linux.oracle.com with the ULN user name and password that you used to register the system.

2. On the Systems tab, click the link named for your system in the list of registered machines.

3. On the System Details page, click **Edit**.

4. On the Edit System Properties page, verify that the **Yum Server** check box is selected and click **Apply Changes**.

5. On the System Details page, click **Manage Subscriptions**.

6. On the System Summary page, select channels from the list of available or subscribed channels and click the arrows to move the channels between the lists.

   Modify the subscribed channels to include Ksplice for Oracle Linux for the system architectures that you want to support as well as any other channels that you want to make available to local systems.

   For example, the following table shows the channels that are available for Ksplice on Oracle Linux.

Channel Name	Channel Label	Description
Ksplice for Oracle Linux 5 (i386)	`ol5_i386_ksplice`	Oracle Ksplice clients, updates, and dependencies for Oracle Linux 5 on i386 systems.
Ksplice for Oracle Linux 5 (x86_64)	`ol5_x86_64_ksplice`	Oracle Ksplice clients, updates, and dependencies for Oracle Linux 5 on x86_64 systems.
Ksplice for Oracle Linux 6 (i386)	`ol6_i386_ksplice`	Oracle Ksplice clients, updates, and dependencies for Oracle Linux 6 on i386 systems.
Ksplice for Oracle Linux 6 (x86_64)	`ol6_x86_64_ksplice`	Oracle Ksplice clients, updates, and dependencies for Oracle Linux 6 on x86_64 systems.
Ksplice for Oracle Linux 7 (x86_64)	`ol7_x86_64_ksplice`	Oracle Ksplice clients, updates, and dependencies for Oracle Linux 7 on x86_64 systems.

   For more information about the release channels that are available, see http://www.oracle.com/technetwork/articles/servers-storage-admin/yum-repo-setup-1659167.html.

7. When you have finished selecting channels, click **Save Subscriptions** and log out of ULN.

3.8.2 Configuring Ksplice Offline Clients

Once you have set up a local yum server that can act as a Ksplice mirror, you can configure your other systems to receive yum and Ksplice updates.

To configure a system as a Ksplice offline client:

1. In the `/etc/yum.repos.d` directory, edit the existing repository file, such as `public-yum-ol6.repo` or `ULN-base.repo`, and disable all entries by setting `enabled=0`.

2. In the `/etc/yum.repos.d` directory, create the file `local-yum.repo`, which contains entries such as the following for an Oracle Linux 6 client:

```
[ol6_x86_64_ksplice]
name=Ksplice for $releasever - $basearch
baseurl=http://local_yum_server/yum/OracleLinux/OL6/ksplice/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=1

[ol6_latest]
name=Oracle Linux $releasever - $basearch - latest
baseurl=http://local_yum_server/yum/OracleLinux/OL6/latest/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=1

[ol6_addons]
name=Oracle Linux $releasever - $basearch - addons
baseurl=http://local_yum_server/yum/OracleLinux/OL6/addons/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0

[ol6_oracle]
name=Oracle Linux $releasever - $basearch - oracle
baseurl=http://local_yum_server/yum/OracleLinux/OL6/oracle/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0

[ol6_ga_base]
name=Oracle Linux $releasever GA - $basearch - base
baseurl=http://local_yum_server/yum/OracleLinux/OL6/0/base/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0

[ol6_u1_base]
name=Oracle Linux $releasever U1 - $basearch - base
baseurl=http://local_yum_server/yum/OracleLinux/OL6/1/base/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0

[ol6_u2_base]
name=Oracle Linux $releasever U2 - $basearch - base
baseurl=http://local_yum_server/yum/OracleLinux/OL6/2/base/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0
```

```
[ol6_u3_base]
name=Oracle Linux $releasever U3 - $basearch - base
baseurl=http://local_yum_server/yum/OracleLinux/OL6/3/base/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0

[ol6_ga_patch]
name=Oracle Linux $releasever GA - $basearch - patch
baseurl=http://local_yum_server/yum/OracleLinux/OL6/0/patch/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0

[ol6_u1_patch]
name=Oracle Linux $releasever U1 - $basearch - patch
baseurl=http://local_yum_server/yum/OracleLinux/OL6/1/patch/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0

[ol6_u2_patch]
name=Oracle Linux $releasever U2 - $basearch - patch
baseurl=http://local_yum_server/yum/OracleLinux/OL6/2/patch/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0

[ol6_u3_patch]
name=Oracle Linux $releasever U3 - $basearch - patch
baseurl=http://local_yum_server/yum/OracleLinux/OL6/3/patch/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=0
```

Replace `local_yum_server` with the IP address or resolvable host name of the local yum server.

In the sample configuration, only the `ol6_latest` and `ol6_x86_64_ksplice` channels are enabled.

> **Note**
>
> As an alternative to specifying a `gpgkey` entry for each repository definition, you can use the following command to import the GPG key:
>
> ```
> # rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY
> ```

3. Install the Ksplice offline client.

```
# yum install uptrack-offline
```

If `yum` cannot connect to the local yum server, check that the firewall settings on that server allow incoming TCP connections to port 80.

4. Install the Ksplice updates that are available for the kernel.

```
# yum install uptrack-updates-`uname -r`
```

For an Oracle Linux 5 client, use this form of the command instead:

```
# yum install uptrack-updates-`uname -r`.`uname -m`
```

As new Ksplice updates are made available, you can use this command to pick up these updates and apply them. It is recommended that you set up a `cron` job to perform this task. For example, the following `crontab` entry for `root` runs the command once per day at 7am:

```
0 7 * * * yum install uptrack-updates-`uname -r`
```

To display information about Ksplice updates, use the `rpm -qa | grep uptrack-updates` and `uptrack-show` commands.

3.9 For More Information About Ksplice Uptrack

You can find out more information about Ksplice Uptrack at http://www.ksplice.com/.

Chapter 4 Boot and Service Configuration

Table of Contents

This chapter describes the Oracle Linux boot process, how to use the GRUB 2 boot loader, how to change the `systemd` target for a system, and how to configure the services that are available for a target.

4.1 About systemd

`systemd` is the new system and service manager in Oracle Linux 7 that replaces the Upstart `init` daemon while providing backward compatibility for legacy Oracle Linux 6 service scripts. `systemd` offers the following benefits over `init`:

- Services are started in parallel wherever possible using socket-based activation and D-Bus.

- Daemons can be started on demand.

- Processes are tracked using control groups (*cgroups*).

- Snapshotting of the system state and restoration of the system state from a snapshot is supported.

- mount points can be configured as `systemd` targets.

`systemd` is the first process that starts after the system boots, and is the final process that is running when the system shuts down. `systemd` controls the final stages of booting and prepares the system for use. `systemd` also speeds up booting by loading services concurrently.

`systemd` allows you to manage various types of units on a system, including services (*name*`.service`) and targets (*name*`.target`), devices (*name*`.device`), file system mount points (*name*`.mount`), and sockets (*name*`.socket`). For example, the following command instructs the system to mount the temporary file system (`tmpfs`) on `/tmp` at boot time:

```
# systemctl enable tmp.mount
```

4.2 About the Boot Process

Understanding the Oracle Linux boot process can help you if you need to troubleshoot problems while booting a system. The boot process involves several files and errors in these files is the usual cause of boot problems.

When an Oracle Linux system boots, it performs the following operations:

1. The computer's BIOS performs a power-on self-test (POST), and then locates and initializes any peripheral devices including the hard disk.

2. The BIOS reads the Master Boot Record (MBR) into memory from the boot device. (For GUID Partition Table (GPT) disks, this MBR is the *protective MBR* on the first sector of the disk.) The MBR stores information about the organization of partitions on that device. On a computer with x86 architecture, the MBR occupies the first 512 bytes of the boot device. The first 446 bytes contain boot code that points to the boot loader program, which can be on the same device or on another device. The next 64 bytes contain the partition table. The final two bytes are the boot signature, which is used for error detection.

   The default boot loader program used on Oracle Linux is GRUB 2, which stands for GRand Unified Bootloader version 2.

3. The boot loader loads the `vmlinuz` kernel image file into memory and extracts the contents of the `initramfs` image file into a temporary, memory-based file system (`tmpfs`).

4. The kernel loads the driver modules from the `initramfs` file system that are needed to access the root file system.

5. The kernel starts the `systemd` process with a process ID of 1 (PID 1). `systemd` is the ancestor of all processes on a system. `systemd` reads its configuration from files in the `/etc/systemd` directory. The `/etc/systemd/system.conf` file controls how `systemd` handles system initialization.

   `systemd` reads the file linked by `/etc/systemd/system/default.target`, for example `/usr/lib/systemd/system/multi-user.target`, to determine the default system target.

   > **Note**
   >
   > You can use a kernel boot parameter to override the default system target. See Section 4.4, "Kernel Boot Parameters".

   The system target file defines the services that `systemd` starts.

   `systemd` brings the system to the state defined by the system target, performing system initialization tasks such as:

   • Setting the host name.

   • Initializing the network.

   • Initializing SELinux based on its configuration.

   • Printing a welcome banner.

   • Initializing the system hardware based on kernel boot arguments.

   • Mounting the file systems, including virtual file systems such as the `/proc` file system.

   • Cleaning up directories in `/var`.

- Starting swapping.

  See Section 4.7, "About System-State Targets".

6. If you have made `/etc/rc.local` executable and you have copied `/usr/lib/systemd/system/rc-local.service` to `/etc/systemd/system`, `systemd` runs any actions that you have defined in `/etc/rc.local`. However, the preferred way of running such local actions is to define your own `systemd` unit.

For information on `systemd` and on how to write `systemd` units, see the `systemd(1)`, `systemd-system.conf(5)`, and `systemd.unit(5)` manual pages.

4.3 About the GRUB 2 Boot Loader

GRUB 2 can load many operating systems in addition to Oracle Linux and it can chain-load proprietary operating systems. GRUB 2 understands the formats of file systems and kernel executables, which allows it to load an arbitrary operating system without needing to know the exact location of the kernel on the boot device. GRUB 2 requires only the file name and drive partitions to load a kernel. You can configure this information by using the GRUB 2 menu or by entering it on the command line.

> **Note**
>
> Do not edit the GRUB 2 configuration file directly. On BIOS-based systems, the configuration file is `/boot/grub2/grub.cfg`. On UEFI-based systems, the configuration file is `/boot/efi/EFI/redhat/grub.cfg`.
>
> The `grub2-mkconfig` command generates the configuration file using the template scripts in `/etc/grub.d` and menu-configuration settings taken from the configuration file, `/etc/default/grub`.

The default menu entry is determined by the value of the `GRUB_DEFAULT` parameter in `/etc/default/grub`. The value `saved` allows you to use the `grub2-set-default` and `grub2-reboot` commands to specify the default entry. `grub2-set-default` sets the default entry for all subsequent reboots and `grub2-reboot` sets the default entry for the next reboot only.

If you specify a numeric value as the value of `GRUB_DEFAULT` or as an argument to either `grub2-reboot` or `grub2-set-default`, GRUB 2 counts the menu entries in the configuration file starting at 0 for the first entry.

To set the UEK as the default boot kernel:

1. Display the menu entries that are defined in the configuration file, for example:

```
# grep '^menuentry' /boot/grub2/grub.cfg
menuentry 'Oracle Linux Everything, with Linux 3.10.0-123.el7.x86_64' ... {
menuentry 'Oracle Linux Everything, with Linux 3.8.13-35.2.1.el7uek.x86_64' ... {
menuentry 'Oracle Linux Everything, with Linux 0-rescue-052e316f566e4a45a3391cff21b4174b' ... {
```

   In this example for a BIOS-based system, the configuration file is `/boot/grub2/grub.cfg`, which contains menu entries 0, 1, and 2 that correspond to the RHCK, UEK, and the rescue kernel respectively.

2. Enter the following commands to make the UEK (entry 1) the default boot kernel:

```
# grub2-set-default 1
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

   Alternatively, you can specify the value of the text of the entry as a string enclosed in quotes.

```
# grub2-set-default 'Oracle Linux Everything, with Linux 3.8.13-35.2.1.el7uek.x86_64'
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

For more information about using, configuring, and customizing GRUB 2, see the GNU GRUB Manual, which is also installed as `/usr/share/doc/grub2-tools-2.00/grub.html`.

4.4 Kernel Boot Parameters

The following table lists commonly-used kernel boot parameters.

Option	Description
`0`, `1`, `2`, `3`, `4`, `5`, or `6`, or `systemd.unit=runlevelN.target`	Specifies the nearest `systemd`-equivalent system-state target to an Oracle Linux 6 run level. `N` can take an integer value between 0 and 6. For a description of system-state targets, see Section 4.7, "About System-State Targets".
`1`, `s`, `S`, `single`, or `systemd.unit=rescue.target`	Specifies the rescue shell. The system boots to single-user mode and does not prompt for the `root` password.
`3` or `systemd.unit=multi-user.target`	Specifies the `systemd` target for multi-user, non-graphical login.
`5` or `systemd.unit=graphical.target`	Specifies the `systemd` target for multi-user, graphical login.
`-b`, `emergency`, or `systemd.unit=emergency.target`	Specifies emergency mode.
`KEYBOARDTYPE=kbtype`	Specifies the keyboard type, which is written to `/etc/sysconfig/keyboard` in the `initramfs`.
`KEYTABLE=kbtype`	Specifies the keyboard layout, which is written to `/etc/sysconfig/keyboard` in the `initramfs`.
`LANG=language_territory.codeset`	Specifies the system language and code set, which is written to `/etc/sysconfig/i18n` in the `initramfs`.
`max_loop=N`	Specifies the number of loop devices (`/dev/loop*`) that are available for accessing files as block devices. The default and maximum values of `N` are 8 and 255.
`nouptrack`	Disables Ksplice Uptrack updates from being applied to the kernel.
`quiet`	Reduces debugging output.
`rd_LUKS_UUID=UUID`	Activates an encrypted Linux Unified Key Setup (LUKS) partition with the specified UUID.
`rd_LVM_VG=vg/lv_vol`	Specifies an LVM volume group and volume to be activated.
`rd_NO_LUKS`	Disables detection of an encrypted LUKS partition.
`rhgb`	Specifies that the Red Hat graphical boot display should be used to indicate the progress of booting.
`rn_NO_DM`	Disables Device-Mapper (DM) RAID detection.

Option	Description
`rn_NO_MD`	Disables Multiple Device (MD) RAID detection.
`ro root=/dev/mapper/vg-lv_root`	Specifies that the root file system is to be mounted read only, and specifies the root file system by the device path of its LVM volume (where `vg` is the name of the volume group).
`rw root=UUID=UUID`	Specifies that the root (`/`) file system is to be mounted read-writable at boot time, and specifies the root partition by its UUID.
`selinux=0`	Disables SELinux.
`SYSFONT=font`	Specifies the console font, which is written to `/etc/sysconfig/i18n` in the `initramfs`.

The kernel boot parameters that were last used to boot a system are recorded in `/proc/cmdline`, for example:

```
# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-3.10.0-123.el7.x86_64 root=UUID=52c1cab6-969f-4872-958d-47f8518267de
ro rootflags=subvol=root vconsole.font=latarcyrheb-sun16 crashkernel=auto vconsole.keymap=uk
rhgb quiet LANG=en_GB.UTF-8
```

For more information, see the `kernel-command-line(7)` manual page.

4.5 Modifying Kernel Boot Parameters Before Booting

To modify boot parameters before booting a kernel:

1. In the GRUB boot menu, use the arrow keys to highlight the required kernel and press the space bar.

   Figure 4.1 shows the GRUB menu with the UEK selected.

   Figure 4.1 GRUB Menu with the UEK selected
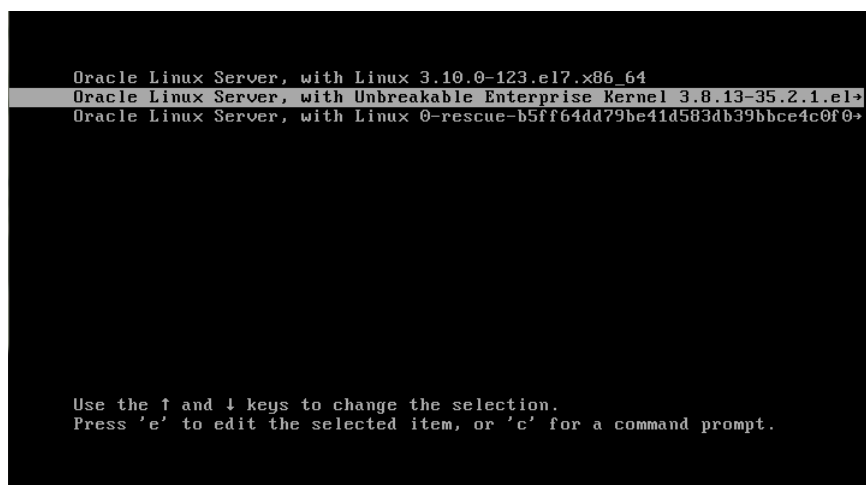
2. Press E to edit the boot configuration for the kernel.

3. Use the arrow keys to scroll down the screen until the cursor is at the start of the boot configuration line for the kernel, which starts `linux16`.

4. Edit the line to change the boot parameters.

For example, press End to go to the end of the line, and enter an additional boot parameter.

Figure 4.2 shows the kernel boot line with the additional parameter `systemd.target=runlevel1.target`, which starts the rescue shell.

Figure 4.2 Kernel Boot Line with an Additional Parameter to Select the Rescue Shell

5. Press Ctrl+X to boot the system.

4.6 Modifying Kernel Boot Parameters in GRUB 2

To modify the boot parameters in the GRUB 2 configuration so that they are applied by default at every reboot:

1. Edit `/etc/default/grub` and modify the parameters in the `GRUB_CMDLINE_LINUX` definition, for example:

```
GRUB_CMDLINE_LINUX="vconsole.font=latarcyrheb-sun16 vconsole.keymap=uk
crashkernel=auto  rd.lvm.lv=ol/swap rd.lvm.lv=ol/root biosdevname=0
rhgb quiet systemd.unit=runlevel3.target"
```

This example adds the parameter `systemd.unit=runlevel3.target` so that the system boots into multi-user, non-graphical mode by default.

2. Rebuild `/boot/grub2/grub.cfg`:

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

The change takes effect for subsequent system reboots of all configured kernels.

4.7 About System-State Targets

`systemd` defines system-state targets allow you to start a system with only the services that are required for a specific purpose. For example, a server can run more efficiently with `multi-user.target`, because it does not run the X Window System at that run level. It is best to perform diagnostics, backups, and upgrades with `rescue.target` when only `root` can use the system. Each run level defines the services that `systemd` stops or starts. For example, `systemd` starts network services for `multi-user.target` and the X Window System for `graphical.target`, whereas it stops both of these services for `rescue.target`.

Table 4.1, "System-State Targets and Equivalent Run-Level Targets" shows the commonly-used system-state targets and their equivalent run-level targets, where compatibility with Oracle Linux 6 run levels is required.

Table 4.1 System-State Targets and Equivalent Run-Level Targets

System-State Targets	Equivalent Run-Level Targets	Description
graphical.target	runlevel5.target	Set up a multi-user system with networking and display manager.
multi-user.target	runlevel2.target	Set up a non-graphical multi-user system with networking.
	runlevel3.target	
	runlevel4.target	
poweroff.target	runlevel0.target	Shut down and power off the system.
reboot.target	runlevel6.target	Shut down and reboot the system.
rescue.target	runlevel1.target	Set up a rescue shell.

The `runlevel*` targets are implemented as symbolic links.

The nearest equivalent `systemd` target to the Oracle Linux 6 run levels 2, 3, and 4 is `multi-user.target`.

For more information, see the `systemd.target(5)` manual page.

4.7.1 Displaying the Default and Active System-State Targets

To display the default system-state target, use the `systemctl get-default` command, for example:

```
# systemctl get-default
graphical.target
```

To display the currently active targets on a system, use the `systemctl list-units` command, for example:

```
# systemctl list-units --type target
UNIT                 LOAD   ACTIVE SUB     DESCRIPTION
basic.target         loaded active active Basic System
cryptsetup.target    loaded active active Encrypted Volumes
getty.target         loaded active active Login Prompts
graphical.target     loaded active active Graphical Interface
local-fs-pre.target  loaded active active Local File Systems (Pre)
local-fs.target      loaded active active Local File Systems
multi-user.target    loaded active active Multi-User System
network.target       loaded active active Network
nfs.target           loaded active active Network File System Server
paths.target         loaded active active Paths
remote-fs.target     loaded active active Remote File Systems
slices.target        loaded active active Slices
sockets.target       loaded active active Sockets
sound.target         loaded active active Sound Card
swap.target          loaded active active Swap
sysinit.target       loaded active active System Initialization
timers.target        loaded active active Timers


LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
```

```
SUB    = The low-level unit activation state, values depend on unit type.

17 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

This sample output for a system with the `graphical` target active shows that this target depends on 16 other active targets, including `network` and `sound` to support networking and sound.

To display the status of all targets on the system, specify the `--all` option:

```
# systemctl list-units --type target --all
UNIT                    LOAD      ACTIVE   SUB    DESCRIPTION
basic.target            loaded active    active Basic System
cryptsetup.target       loaded active    active Encrypted Volumes
emergency.target        loaded inactive dead    Emergency Mode
final.target            loaded inactive dead    Final Step
getty.target            loaded active    active Login Prompts
graphical.target        loaded active    active Graphical Interface
local-fs-pre.target     loaded active    active Local File Systems (Pre)
local-fs.target         loaded active    active Local File Systems
multi-user.target       loaded active    active Multi-User System
network-online.target   loaded inactive dead    Network is Online
network.target          loaded active    active Network
nfs.target              loaded active    active Network File System Server
nss-lookup.target       loaded inactive dead    Host and Network Name Lookups
nss-user-lookup.target  loaded inactive dead    User and Group Name Lookups
paths.target            loaded active    active Paths
remote-fs-pre.target    loaded inactive dead    Remote File Systems (Pre)
remote-fs.target        loaded active    active Remote File Systems
rescue.target           loaded inactive dead    Rescue Mode
shutdown.target         loaded inactive dead    Shutdown
slices.target           loaded active    active Slices
sockets.target          loaded active    active Sockets
sound.target            loaded active    active Sound Card
swap.target             loaded active    active Swap
sysinit.target          loaded active    active System Initialization
syslog.target           not-found inactive dead   syslog.target
time-sync.target        loaded inactive dead    System Time Synchronized
timers.target           loaded active    active Timers
umount.target           loaded inactive dead    Unmount All Filesystems

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

28 loaded units listed.
To show all installed unit files use 'systemctl list-unit-files'.
```

For more information, see the `systemctl(1)` and `systemd.target(5)` manual pages.

4.7.2 Changing the Default and Active System-State Targets

Use the `systemctl set-default` command to change the default system-state target, for example:

```
# systemctl set-default multi-user.target
rm '/etc/systemd/system/default.target'
ln -s '/usr/lib/systemd/system/multi-user.target' '/etc/systemd/system/default.target'
```

> **Note**
>
> This command changes the target to which the default target is linked, but does not change the state of the system.

To change the currently active system target, use the `systemctl isolate` command, for example:

```
# systemctl isolate multi-user.target
```

Listing all targets shows that `graphical` and `sound` targets are not active:

```
# systemctl list-units --type target --all
UNIT                    LOAD   ACTIVE   SUB   DESCRIPTION
basic.target            loaded active   active Basic System
cryptsetup.target       loaded active   active Encrypted Volumes
emergency.target        loaded inactive dead   Emergency Mode
final.target            loaded inactive dead   Final Step
getty.target            loaded active   active Login Prompts
graphical.target        loaded inactive dead   Graphical Interface
local-fs-pre.target     loaded active   active Local File Systems (Pre)
local-fs.target         loaded active   active Local File Systems
multi-user.target       loaded active   active Multi-User System
network-online.target   loaded inactive dead   Network is Online
network.target          loaded active   active Network
nfs.target              loaded active   active Network File System Server
nss-lookup.target       loaded inactive dead   Host and Network Name Lookups
nss-user-lookup.target  loaded inactive dead   User and Group Name Lookups
paths.target            loaded active   active Paths
remote-fs-pre.target    loaded inactive dead   Remote File Systems (Pre)
remote-fs.target        loaded active   active Remote File Systems
rescue.target           loaded inactive dead   Rescue Mode
shutdown.target         loaded inactive dead   Shutdown
slices.target           loaded active   active Slices
sockets.target          loaded active   active Sockets
sound.target            loaded inactive dead   Sound Card
swap.target             loaded active   active Swap
sysinit.target          loaded active   active System Initialization
syslog.target           not-found inactive dead   syslog.target
time-sync.target        loaded inactive dead   System Time Synchronized
timers.target           loaded active   active Timers
umount.target           loaded inactive dead   Unmount All Filesystems

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

28 loaded units listed.
To show all installed unit files use 'systemctl list-unit-files'.
```

For more information, see the `systemctl(1)` manual page.

4.7.3 Shutting Down, Suspending, or Rebooting the System

Table 4.2, "systemctl Commands for Shutting Down, Suspending, or Rebooting a System" shows the
`systemctl` commands for shutting down rebooting, or otherwise suspending the operation of a system.

Table 4.2 systemctl Commands for Shutting Down, Suspending, or Rebooting a System

systemctl Command	Description
systemctl halt	Halt the system.
systemctl hibernate	Put the system into hibernation.
systemctl hybrid-sleep	Put the system into hibernation and suspend its operation.
systemctl poweroff	Halt and power off the system.
systemctl reboot	Reboot the system.
systemctl suspend	Suspend the system.

For more information, see the `systemctl(1)` manual page.

4.7.4 Starting and Stopping Services

To start a service, use the `systemctl` command with the `start` argument, for example:

```
# systemctl start sshd
```

For legacy scripts in `/etc/init.d` that have not been ported as `systemd` services, you can run the script directly with the `start` argument:

```
# /etc/init.d/yum-cron start
```

To stop a service, use the `stop` argument to `systemctl`:

```
# systemctl stop sshd
```

> **Note**
>
> Changing the state of a service only lasts as long as the system remains at the same state. If you stop a service and then change the system-state target to one in which the service is configured to run (for example, by rebooting the system), the service restarts. Similarly, starting a service does not enable the service to start following a reboot. See Section 4.7.5, "Enabling and Disabling Services".

`systemctl` supports the `disable`, `enable`, `reload`, `restart`, `start`, `status`, and `stop` actions for services. For other actions, you must either run the script that the service provides to support these actions, or for legacy scripts, the `/etc/init.d` script with the required action argument. For legacy scripts, omitting the argument to the script displays a usage message, for example:

```
# /etc/init.d/yum-cron
Usage: /etc/init.d/yum-cron {start|stop|status|restart|reload|force-reload|condrestart}
```

For more information, see the `systemctl(1)` manual page.

4.7.5 Enabling and Disabling Services

You can use the `systemctl` command to enable or disable a service from starting when the system starts, for example:

```
# systemctl enable httpd
ln -s '/usr/lib/systemd/system/httpd.service' \
  '/etc/systemd/system/multi-user.target.wants/httpd.service'
```

The command enables a service by creating a symbolic link for the lowest-level system-state target at which the service should start. In the example, the command creates the symbolic link `httpd.service` for the `multi-user` target.

Disabling a service removes the symbolic link:

```
# systemctl disable httpd
rm '/etc/systemd/system/multi-user.target.wants/httpd.service'
```

You can use the `is-enabled` subcommand to check whether a service is enabled:

```
# systemctl is-enabled httpd
disabled
# systemctl is-enabled nfs
```

```
enabled
```

For more information, see the `systemctl(1)` manual page.

4.7.6 Displaying the Status of Services

You can use the `is-active` subcommand to check whether a service is running (*active*) or not running (*inactive*):

```
# systemctl is-active httpd
active
# systemctl is-active nfs
inactive
```

You can use the `status` action to view a detailed summary of the status of a service, including a tree of all the tasks in the *control group* (cgroup) that the service implements:

```
# systemctl status httpd
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled)
   Active: active (running) since Mon 2014-04-28 15:02:40 BST; 1s ago
 Main PID: 6452 (httpd)
   Status: "Processing requests..."
   CGroup: /system.slice/httpd.service
           ├─6452 /usr/sbin/httpd -DFOREGROUND
           ├─6453 /usr/sbin/httpd -DFOREGROUND
           ├─6454 /usr/sbin/httpd -DFOREGROUND
           ├─6455 /usr/sbin/httpd -DFOREGROUND
           ├─6456 /usr/sbin/httpd -DFOREGROUND
           └─6457 /usr/sbin/httpd -DFOREGROUND

Apr 28 15:02:40 localhost.localdomain systemd[1]: Started The Apache HTTP Ser...
Hint: Some lines were ellipsized, use -l to show in full.
```

A cgroup is a collection of processes that are bound together so that you can control their access to system resources. In the example, the cgroup for the `httpd` service is `httpd.service`, which is in the `system` *slice.*

Slices divide the cgroups on a system into different categories. To display the slice and cgroup hierarchy, use the `systemd-cgls` command:

```
# systemd-cgls
├─user.slice
│ ├─user-1000.slice
│ │ └─session-12.scope
│ │   ├─3152 gdm-session-worker [pam/gdm-password]
│ │   ├─3169 /usr/bin/gnome-keyring-daemon --daemonize --login
│ │   ├─3171 gnome-session --session gnome-classic
│ │   │ ...
│ │   └─3763 /usr/libexec/evolution-calendar-factory
│ └─user-0.slice
│   ├─session-13.scope
│   │ ├─3810 sshd: root@pts/0
│   │ ├─3836 -bash
│   │ ├─4015 systemd-cgls
│   │ └─4016 systemd-cgls
│   └─session-6.scope
│     └─3030 /usr/sbin/anacron -s
└─system.slice
  ├─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 23
  ├─bluetooth.service
```

```
 │    └─3421 /usr/sbin/bluetoothd -n
 ├─udisks2.service
 │    └─3420 /usr/lib/udisks2/udisksd --no-debug
 ├─colord.service
 │    └─2812 /usr/libexec/colord
 ├─upower.service
 │    └─2760 /usr/libexec/upowerd
 ├─iscsid.service
 │    ├─1288 /usr/sbin/iscsid
 │    └─1289 /usr/sbin/iscsid
 │ ...
 ├─dbus.service
 │    └─427 /bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --sy
 ├─firewalld.service
 │    └─391 /usr/bin/python /usr/sbin/firewalld --nofork --nopid
 ├─iprdump.service
 │    └─524 /sbin/iprdump --daemon
 ├─iprinit.service
 │    └─466 /sbin/iprinit --daemon
 ├─iprupdate.service
 │    └─467 /sbin/iprupdate --daemon
 └─network.service
      └─736 /sbin/dhclient -H localhost -1 -q -lf /var/lib/dhclient/dhclient-f174a
```

`system.slice` contains services and other system processes. `user.slice` contains user processes, which run within transient cgroups called *scopes*. In the example, the processes for the user with ID 1000 are running in the scope `session-12.scope` under the slice `/user.slice/user-1000.slice`.

You can use the `systemctl` command to limit the CPU, I/O, memory, and other resources that are available to the processes in service and scope cgroups. See Section 4.7.7, "Controlling Access to System Resources".

For more information, see the `systemctl(1)` and `systemd-cgls(1)` manual pages.

4.7.7 Controlling Access to System Resources

You can use the `systemctl` command to control a cgroup's access to system resources, for example:

```
# systemctl set-property httpd.service CPUShares=512 MemoryLimit=1G
```

`CPUShare` controls access to CPU resources. As the default value is 1024, a value of 512 halves the access that the processes in the cgroup have to CPU time. Similarly, `MemoryLimit` controls the maximum amount of memory that the cgroup can use.

Note

You do not need to specify the `.service` extension to the name of a service.

If you specify the `--runtime` option, the setting does not persist across system reboots.

```
# systemctl --runtime set-property httpd CPUShares=512 MemoryLimit=1G
```

Alternatively, you can change the resource settings for a service under the `[Service]` heading in the service's configuration file in `/usr/lib/systemd/system`. After editing the file, make `systemd` reload its configuration files and then restart the service:

```
# systemctl daemon-reload
# systemctl restart service
```

You can run general commands within scopes and use `systemctl` to control the access that these transient cgroups have to system resources. To run a command within in a scope, use the `systemd-run` command:

```
# systemd-run --scope --unit=group_name [--slice=slice_name] command
```

If you do not want to create the group under the default `system` slice, you can specify another slice or the name of a new slice.

> **Note**
>
> If you do not specify the `--scope` option, the control group is a created as a service rather than as a scope.

For example, run a command named `mymonitor` in `mymon.scope` under `myslice.slice`:

```
# systemd-run --scope --unit=mymon --slice=myslice mymonitor
Running as unit mymon.scope.
```

You can then use `systemctl` to control the access that a scope has to system resources in the same way as for a service. However, unlike a service, you must specify the `.scope` extension, for example:

```
# systemctl --runtime set-property mymon.scope CPUShares=256
```

For more information see the `systemctl(1)`, `systemd-cgls(1)`, and `systemd.resource-control(5)` manual pages.

4.7.8 Modifying systemd Configuration Files

If you want to change the configuration of `systemd`, copy the `service`, `target`, `mount`, `socket` or other file from `/usr/lib/systemd/system` to `/etc/systemd/system` and edit this copy of the original file. The version of the file in `/etc/systemd/system` takes precedence over the version in `/usr/lib/systemd/system`, and is not overwritten when you update a package that touches files in `/usr/lib/systemd/system`. To make `systemd` revert to using the original version of the file, either rename or delete the modified copy of the file in `/etc/systemd/system`.

4.7.9 Running systemctl on a Remote System

If the `sshd` service is running on a remote Oracle Linux 7 system, you can use the `-H` option with `systemctl` to control the system remotely, as shown in this example:

```
# systemctl -H root@10.0.0.2 status sshd
root@10.0.0.2's password: password
sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled)
   Active: active (running) since Fri 2014-05-23 09:27:22 BST; 5h 43min ago
  Process: 1498 ExecStartPre=/usr/sbin/sshd-keygen (code=exited, status=0/SUCCESS)
 Main PID: 1524 (sshd)
   CGroup: /system.slice/sshd.service
```

For more information see the `systemctl(1)` manual page.

Chapter 5 System Configuration Settings

Table of Contents

This chapter describes the files and virtual file systems that you can use to change configuration settings for your system.

5.1 About /etc/sysconfig Files

The `/etc/sysconfig` directory contains files that control your system's configuration. The contents of this directory depend on the packages that you have installed on your system.

Some of the files that you might find in the `/etc/sysconfig` directory include:

`atd`	Specifies additional command line arguments for the `atd` daemon.
`authconfig`	Specifies whether various authentication mechanisms and options may be used. For example, the entry `USEMKHOMEDIR=no` disables the creation of a home directory for a user when he or she first logs in.
`autofs`	Defines custom options for automatically mounting devices and controlling the operation of the automounter.
`crond`	Passes arguments to the `crond` daemon at boot time.
`firewalld`	Passes arguments to the firewall daemon (`firewalld`) at boot time.
`grub`	Specifies default settings for the GRUB 2 boot loader. This file is a symbolic link to `/etc/default/grub`. For more information, see Section 4.3, "About the GRUB 2 Boot Loader".
`init`	Controls how the system appears and functions during the boot process.
`keyboard`	Specifies the keyboard.
`modules` (directory)	Contains scripts that the kernel runs to load additional modules at boot time. A script in the `modules` directory must have the extension `.modules` and it must have 755 executable permissions. For an example, see the `bluez-uinput.modules` script that loads the `uinput` module. For more information, see Section 6.5, "Specifying Modules to be Loaded at Boot Time".
`named`	Passes arguments to the name service daemon at boot time. The `named` daemon is a Domain Name System (DNS) server that is part of the Berkeley Internet Name Domain (BIND) distribution. This server maintains a table that associates host names with IP addresses on the network.

nfs   Controls which ports remote procedure call (RPC) services use for NFS v2 and v3. This file allows you to set up firewall rules for NFS v2 and v3. Firewall configuration for NFS v4 does not require you to edit this file.

ntpd   Passes arguments to the network time protocol (NTP) daemon at boot time.

samba   Passes arguments to the `smbd`, `nmbd`, and `winbindd` daemons at boot time to support file-sharing connectivity for Windows clients, NetBIOS-over-IP naming service, and connection management to domain controllers.

selinux   Controls the state of SELinux on the system. This file is a symbolic link to `/etc/selinux/config`. For more information, see Section 26.2.3, "Setting SELinux Modes".

snapper   Defines a list of btrfs file systems and thinly-provisioned LVM volumes whose contents can be recorded as snapshots by the `snapper` utility. For more information, see Section 21.7.1, "Using snapper with Btrfs Subvolumes" and Section 19.3.6, "Using snapper with Thinly-Provisioned Logical Volumes".

sysstat   Configures logging parameters for system activity data collector utilities such as `sadc`.

For more information, see `/usr/share/doc/initscripts*/sysconfig.txt`.

> **Note**
>
> In previous release of Oracle Linux, the host name of the system was defined in `/etc/sysconfig/network`. The host name is now defined in `/etc/hostname` and can be changed by using the `hostnamectl` command. System-wide default localization settings such as the default language, keyboard, and console font were defined in `/etc/sysconfig/i18n`. These settings are now defined in `/etc/locale.conf` and `/etc/vconsole.conf`.
>
> For more information, see the `hostname(5)`, `hostnamectl(1)`, `locale.conf(5)`, and `vconsole.conf(5)` manual pages.

5.2 About the /proc Virtual File System

The files in the `/proc` directory hierarchy contain information about your system hardware and the processes that are running on the system. You can change the configuration of the kernel by writing to certain files that have write permission.

The name of the `proc` file system stems from its original purpose on the Oracle Solaris operating system, which was to allow access by debugging tools to the data structures inside running processes. Linux added this interface and extended it to allow access to data structures in the kernel. Over time, `/proc` became quite disordered and the `sysfs` file system was created in an attempt to tidy it up. For more information, see Section 5.3, "About the /sys Virtual File System".

Files under the `/proc` directory are virtual files that the kernel creates on demand to present a browsable view of the underlying data structures and system information. As such, `/proc` is an example of a virtual file system. Most virtual files are listed as zero bytes in size, but they contain a large amount of information when viewed.

Virtual files such as `/proc/interrupts`, `/proc/meminfo`, `/proc/mounts`, and `/proc/partitions` provide a view of the system's hardware. Others, such as `/proc/filesystems` and the files under `/proc/sys` provide information about the system's configuration and allow this configuration to be modified.

Files that contain information about related topics are grouped into virtual directories. For example, a separate directory exists in /proc for each process that is currently running on the system, and the directory's name corresponds to the numeric process ID. /proc/1 corresponds to the systemd process, which has a PID of 1.

You can use commands such as cat, less, and view to examine virtual files within /proc. For example, /proc/cpuinfo contains information about the system's CPUs:

```
# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 42
model name      : Intel(R) Core(TM) i5-2520M CPU @ 2.50GHz
stepping        : 7
cpu MHz         : 2393.714
cache size      : 6144 KB
physical id     : 0
siblings        : 2
core id         : 0
cpu cores       : 2
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 5
wp              : yes
...
```

Certain files under /proc require root privileges for access or contain information that is not human-readable. You can use utilities such as lspci, free, and top to access the information in these files. For example, lspci lists all PCI devices on a system:

```
# lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:02.0 VGA compatible controller: InnoTek Systemberatung GmbH VirtualBox Graphics Adapter
00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet Controller (rev 02)
00:04.0 System peripheral: InnoTek Systemberatung GmbH VirtualBox Guest Service
00:05.0 Multimedia audio controller: Intel Corporation 82801AA AC'97 Audio Controller (rev 01)
00:06.0 USB controller: Apple Inc. KeyLargo/Intrepid USB
00:07.0 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:0b.0 USB controller: Intel Corporation 82801FB/FBM/FR/FW/FRW (ICH6 Family) USB2 EHCI Controller
00:0d.0 SATA controller: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [AHCI mode]
        (rev 02)
...
```

5.2.1 Virtual Files and Directories Under /proc

The following table lists the most useful virtual files and directories under the /proc directory hierarchy.

Table 5.1 Useful Virtual Files and Directories Under /proc

Virtual File or Directory	Description
PID (Directory)	Provides information about the process with the process ID (PID). The directory's owner and group is same as the process's. Useful files under the directory include:
	cmdline Command path.
	cwd Symbolic link to the process's current working directory.

Virtual File or Directory	Description
	`environ` Environment variables.
	`exe` Symbolic link to the command executable.
	`fd`/*N* File descriptors.
	`maps` Memory maps to executable and library files.
	`root` Symbolic link to the effective root directory for the process.
	`stack` The contents of the kernel stack.
	`status` Run state and memory usage.
`buddyinfo`	Provides information for diagnosing memory fragmentation.
`bus` (directory)	Contains information about the various buses (such as `pci` and `usb`) that are available on the system. You can use commands such as `lspci`, `lspcmcia`, and `lsusb` to display information for such devices.
`cgroups`	Provides information about the resource control groups that are in use on the system.
`cmdline`	Lists parameters passed to the kernel at boot time.
`cpuinfo`	Provides information about the system's CPUs.
`crypto`	Provides information about all installed cryptographic cyphers.
`devices`	Lists the names and major device numbers of all currently configured characters and block devices.
`dma`	Lists the direct memory access (DMA) channels that are currently in use.
`driver` (directory)	Contains information about drivers used by the kernel, such as those for non-volatile RAM (`nvram`), the real-time clock (`rtc`), and memory allocation for sound (`snd-page-alloc`).
`execdomains`	Lists the execution domains for binaries that the Oracle Linux kernel supports.
`filesystems`	Lists the file system types that the kernel supports. Entries marked with `nodev` are not in use.
`fs` (directory)	Contains information about mounted file systems, organized by file system type.
`interrupts`	Records the number of interrupts per interrupt request queue (IRQ) for each CPU since system startup.
`iomem`	Lists the system memory map for each physical device.
`ioports`	Lists the range of I/O port addresses that the kernel uses with devices.
`irq` (directory)	Contains information about each IRQ. You can configure the affinity between each IRQ and the system CPUs.
`kcore`	Presents the system's physical memory in `core` file format that you can examine using a debugger such as `crash` or `gdb`. This file is not human-readable.

Virtual File or Directory	Description
kmsg	Records kernel-generated messages, which are picked up by programs such as dmesg.
loadavg	Displays the system load averages (number of queued processes) for the past 1, 5, and 15 minutes, the number of running processes, the total number of processes, and the PID of the process that is running.
locks	Displays information about the file locks that the kernel is currently holding on behalf of processes. The information provided includes: • lock class (FLOCK or POSIX) • lock type (ADVISORY or MANDATORY) • access type (READ or WRITE) • process ID • major device, minor device, and inode numbers • bounds of the locked region
mdstat	Lists information about multiple-disk RAID devices.
meminfo	Reports the system's usage of memory in more detail than is available using the free or top commands.
modules	Displays information about the modules that are currently loaded into the kernel. The lsmod command formats and displays the same information, excluding the kernel memory offset of a module.
mounts	Lists information about all mounted file systems.
net (directory)	Provides information about networking protocol, parameters, and statistics. Each directory and virtual file describes aspects of the configuration of the system's network.
partitions	Lists the major and minor device numbers, number of blocks, and name of partitions mounted by the system.
scsi/device_info	Provides information about supported SCSI devices.
scsi/scsi and scsi/sg/*	Provide information about configured SCSI devices, including vendor, model, channel, ID, and LUN data .
self	Symbolic link to the process that is examining /proc.
slabinfo	Provides detailed information about slab memory usage.
softirqs	Displays information about software interrupts (*softirqs*). A softirq is similar to a hardware interrupt (*hardirq*) and allow the kernel to perform asynchronous processing that would take too long during a hardware interrupt.
stat	Records information about the system since it was started, including: cpu — Total CPU time (measured in *jiffies*) spent in user mode, low-priority user mode, system mode, idle, waiting for I/O, handling hardirq events, and handling softirq events. cpu$N$ — Times for CPU N.

Virtual File or Directory	Description
swaps	Provides information on swap devices. The units of size and usage are kilobytes.
sys (directory)	Provides information about the system and also allows you to enable, disable, or modify kernel features. You can write new settings to any file that has write permission. See Section 5.2.2, "Changing Kernel Parameters".
	The following subdirectory hierarchies of /proc/sys contain virtual files, some of whose values you can usefully alter:
	dev Device parameters.
	fs File system parameters.
	kernel Kernel configuration parameters.
	net Networking parameters.
sysvipc (directory)	Provides information about the usage of System V Interprocess Communication (IPC) resources for messages (msg), semaphores (sem), and shared memory (shm).
tty (directory)	Provides information about the available and currently used terminal devices on the system. The drivers virtual file lists the devices that are currently configured.
vmstat	Provides information about virtual memory usage.

For more information, see the proc(5) manual page.

5.2.2 Changing Kernel Parameters

Some virtual files under /proc, and under /proc/sys in particular, are writable and you can use them to adjust settings in the kernel. For example, to change the host name, you can write a new value to /proc/sys/kernel/hostname:

```
# echo www.mydomain.com > /proc/sys/kernel/hostname
```

Other files take value that take binary or Boolean values. For example, the value of /proc/sys/net/ipv4/ip_forward determines whether the kernel forwards IPv4 network packets.

```
# cat /proc/sys/net/ipv4/ip_forward
0
# echo 1 > /proc/sys/net/ipv4/ip_forward
# cat /proc/sys/net/ipv4/ip_forward
1
```

You can use the sysctl command to view or modify values under the /proc/sys directory.

> **Note**
>
> Even root cannot bypass the file access permissions of virtual file entries under /proc. If you attempt to change the value of a read-only entry such as /proc/partitions, there is no kernel code to service the write() system call.

To display all of the current kernel settings:

```
# sysctl -a
kernel.sched_child_runs_first = 0
kernel.sched_min_granularity_ns = 2000000
kernel.sched_latency_ns = 10000000
kernel.sched_wakeup_granularity_ns = 2000000
kernel.sched_shares_ratelimit = 500000
...
```

> **Note**
>
> The delimiter character in the name of a setting is a period (`.`) rather than a slash (`/`) in a path relative to `/proc/sys`. For example, `net.ipv4.ip_forward` represents `net/ipv4/ip_forward` and `kernel.msgmax` represents `kernel/msgmax`.

To display an individual setting, specify its name as the argument to `sysctl`:

```
# sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0
```

To change the value of a setting, use the following form of the command:

```
# sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

Changes that you make in this way remain in force only until the system is rebooted. To make configuration changes persist after the system is rebooted, you must add them to the `/etc/sysctl.conf` file. Any changes that you make to this file take effect when the system reboots or if you run the `sysctl -p` command, for example:

```
# sed -i '/net.ipv4.ip_forward/s/= 0/= 1/' /etc/sysctl.conf
# grep ip_forward /etc/sysctl.conf
net.ipv4.ip_forward = 1
# sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0
# sysctl -p
net.ipv4.ip_forward = 1
net.ipv4.conf.default.rp_filter = 1
...
kernel.shmall = 4294967296
# sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 1
```

For more information, see the `sysctl(8)` and `sysctl.conf(5)` manual pages.

5.2.3 Parameters that Control System Performance

The following parameters control aspects of system performance:

fs.file-max

Specifies the maximum number of open files for all processes. Increase the value of this parameter if you see messages about running out of file handles.

net.core.netdev_max_backlog

Specifies the size of the receiver backlog queue, which is used if an interface receives packets faster than the kernel can process them. If this queue is too small, packets are lost at the receiver, rather than on the network.

net.core.rmem_max

Specifies the maximum read socket buffer size. To minimize network packet loss, this buffer must be large enough to handle incoming network packets.

net.core.wmem_max

Specifies the maximum write socket buffer size. To minimize network packet loss, this buffer must be large enough to handle outgoing network packets.

net.ipv4.tcp_available_congestion_control

Displays the TCP congestion avoidance algorithms that are available for use. Use the `modprobe` command if you need to load additional modules such as `tcp_htcp` to implement the `htcp` algorithm.

net.ipv4.tcp_congestion_control

Specifies which TCP congestion avoidance algorithm is used.

net.ipv4.tcp_max_syn_backlog

Specifies the number of outstanding `SYN` requests that are allowed. Increase the value of this parameter if you see `synflood` warnings in your logs, and investigation shows that they are occurring because the server is overloaded by legitimate connection attempts.

net.ipv4.tcp_rmem

Specifies minimum, default, and maximum receive buffer sizes that are used for a TCP socket. The maximum value cannot be larger than `net.core.rmem_max`.

net.ipv4.tcp_wmem

Specifies minimum, default, and maximum send buffer sizes that are used for a TCP socket. The maximum value cannot be larger than `net.core.wmem_max`.

vm.swappiness

Specifies how likely the kernel is to write loaded pages to swap rather than drop pages from the system page cache. When set to 0, swapping only occurs to avoid an out of memory condition. When set to 100, the kernel swaps aggressively. For a desktop system, setting a lower value can improve system responsiveness by decreasing latency. The default value is 60.

> **⚠ Caution**
>
> This parameter is intended for use with laptops to reduce power consumption by the hard disk. Do not adjust this value on server systems.

5.2.4 Parameters that Control Kernel Panics

The following parameters control the circumstances under which a kernel panic can occur:

kernel.hung_task_panic

(UEK R3 only) If set to 1, the kernel panics if any kernel or user thread sleeps in the `TASK_UNINTERRUPTIBLE` state (*D state*) for more than `kernel.hung_task_timeout_secs` seconds.

A process remains in D state while waiting for I/O to complete. You cannot kill or interrupt a process in this state.

The default value is 0, which disables the panic.

> **Tip**
>
> To diagnose a hung thread, you can examine `/proc/`*`PID`*`/stack`, which displays the kernel stack for both kernel and user threads.

kernel.hung_task_timeout_secs

(UEK R3 only) Specifies how long a user or kernel thread can remain in D state before a warning message is generated or the kernel panics (if the value of `kernel.hung_task_panic` is 1). The default value is 120 seconds. A value of 0 disables the timeout.

kernel.nmi_watchdog

If set to 1 (default), enables the non-maskable interrupt (NMI) watchdog thread in the kernel. If you want to use the NMI switch or the OProfile system profiler to generate an undefined NMI, set the value of `kernel.nmi_watchdog` to 0.

kernel.panic

Specifies the number of seconds after a panic before a system will automatically reset itself.

If the value is 0, the system hangs, which allows you to collect detailed information about the panic for troubleshooting. This is the default value.

To enable automatic reset, set a non-zero value. If you require a memory image (`vmcore`), allow enough time for Kdump to create this image. The suggested value is 30 seconds, although large systems will require a longer time.

kernel.panic_on_io_nmi

If set to 0 (default), the system tries to continue operations if the kernel detects an I/O channel check (IOCHK) NMI that usually indicates a uncorrectable hardware error. If set to 1, the system panics.

kernel.panic_on_oops

If set to 0, the system tries to continue operations if the kernel encounters an oops or BUG condition. If set to 1 (default), the system delays a few seconds to give the kernel log daemon, `klogd`, time to record the oops output before the panic occurs.

In an OCFS2 cluster. set the value to 1 to specify that a system must panic if a kernel oops occurs. If a kernel thread required for cluster operation crashes, the system must reset itself. Otherwise, another node might not be able to tell whether a node is slow to respond or unable to respond, causing cluster operations to hang.

kernel.panic_on_stackoverflow

(RHCK only) If set to 0 (default), the system tries to continue operations if the kernel detects an overflow in a kernel stack. If set to 1, the system panics.

kernel.panic_on_unrecovered_nmi

If set to 0 (default), the system tries to continue operations if the kernel detects an NMI that usually indicates an uncorrectable parity or ECC memory error. If set to 1, the system panics.

kernel.softlockup_panic

If set to 0 (default), the system tries to continue operations if the kernel detects a *soft-lockup* error that causes the NMI watchdog thread to fail to update its time stamp for more than twice the value of `kernel.watchdog_thresh` seconds. If set to 1, the system panics.

kernel.unknown_nmi_panic

If set to 1, the system panics if the kernel detects an undefined NMI. You would usually generate an undefined NMI by manually pressing an NMI switch. As the NMI watchdog thread also uses the undefined NMI, set the value of `kernel.unknown_nmi_panic` to 0 if you set `kernel.nmi_watchdog` to 1.

kernel.watchdog_thresh

Specifies the interval between generating an NMI performance monitoring interrupt that the kernel uses to check for *hard-lockup* and *soft-lockup* errors. A hard-lockup error is assumed if a CPU is unresponsive to the interrupt for more than `kernel.watchdog_thresh` seconds. The default value is 10 seconds. A value of 0 disables the detection of lockup errors.

vm.panic_on_oom

If set to 0 (default), the kernel's OOM-killer scans through the entire task list and attempts to kill a memory-hogging process to avoid a panic. If set to 1, the kernel panics but can survive under certain conditions. If a process limits allocations to certain nodes by using memory policies or cpusets, and those nodes reach memory exhaustion status, the OOM-killer can kill one process. No panic occurs in this case because other nodes' memory might be free and the system as a whole might not yet be out of memory. If set to 2, the kernel always panics when an OOM condition occurs. Settings of 1 and 2 are for intended for use with clusters, depending on your preferred failover policy.

5.3 About the **/sys** Virtual File System

In addition to `/proc`, the kernel exports information to the `/sys` virtual file system (`sysfs`). Programs such as the dynamic device manager, `udev`, use `/sys` to access device and device driver information. The implementation of `/sys` has helped to tidy up the `/proc` file system as most hardware information has been moved to `/sys`.

> **Note**
>
> `/sys` exposes kernel data structures and control points, which implies that it might contain circular references, where a directory links to an ancestor directory. As a result, a `find` command used on `/sys` might never terminate.

5.3.1 Virtual Directories Under **/sys**

The following table lists useful virtual directories under the `/sys` directory hierarchy.

Table 5.2 Useful Virtual Directories Under `/sys`

Virtual Directory	Description
block	Contains subdirectories for block devices. For example: `/sys/block/sda`.
bus	Contains subdirectories for each supported physical bus type, such as `pci`, `pcmcia`, `scsi`, or `usb`. Under each bus type, the `devices` directory lists discovered devices, and the `drivers` directory contains directories for each device driver.

Virtual Directory	Description
`class`	Contains subdirectories for every class of device that is registered with the kernel.
`devices`	Contains the global device hierarchy of all devices on the system. The platform directory contains peripheral devices such as device controllers that are specific to a particular platform. The `system` directory contains non-peripheral devices such as CPUs and APICs. The `virtual` directory contains virtual and pseudo devices. See Chapter 7, *Device Management*.
`firmware`	Contains subdirectories for firmware objects.
`module`	Contains subdirectories for each module loaded into the kernel. You can alter some parameter values for loaded modules. See Section 6.4, "About Module Parameters".
`power`	Contains attributes that control the system's power state.

For more information, see https://www.kernel.org/doc/Documentation/filesystems/sysfs.txt.

Chapter 6 Kernel Modules

Table of Contents

This chapter describes how to load, unload, and modify the behavior of kernel modules.

6.1 About Kernel Modules

The boot loader loads the kernel into memory. You can add new code to the kernel by including the source files in the kernel source tree and recompiling the kernel. Kernel modules, which can be dynamically loaded and unloaded on demand, provide device drivers that allow the kernel to access new hardware, support different file system types, and extend its functionality in other ways. To avoid wasting memory on unused device drivers, Oracle Linux supports loadable kernel modules (LKMs), which allow a system to run with only the device drivers and kernel code that it requires loaded into memory.

6.2 Listing Information about Loaded Modules

Use the `lsmod` command to list the modules that are currently loaded into the kernel.

```
# lsmod
Module                  Size  Used by
nls_utf8                1405  1
fuse                   59164  0
tun                    12079  0
autofs4                22739  3
...
ppdev                   7901  0
parport_pc             21262  0
parport                33812  2 ppdev,parport_pc
...
```

Note

This command produces its output by reading the `/proc/modules` file.

The output shows the module name, the amount of memory it uses, the number of processes using the module and the names of other modules on which it depends. In the sample output, the module `parport` depends on the modules `ppdev` and `parport_pc`, which are loaded in advance of `parport`. Two processes are currently using all three modules.

To display detailed information about a module, use the `modinfo` command, for example:

```
# modinfo ahci
filename:       /lib/modules/2.6.32-300.27.1.el6uek.x86_64/kernel/drivers/ata/ahci.ko
version:        3.0
license:        GPL
description:    AHCI SATA low-level driver
author:         Jeff Garzik
srcversion:     AC5EC885397BF332DE16389
```

```
alias:          pci:v*d*sv*sd*bc01sc06i01*
...
depends:
vermagic:       2.6.32-300.27.1.el6uek.x86_64 SMP mod_unload modversions
parm:           skip_host_reset:skip global host reset (0=don't skip, 1=skip) (int)
parm:           ignore_sss:Ignore staggered spinup flag (0=don't ignore, 1=ignore) (int)
...
```

The output includes the following information:

filename   Absolute path of the kernel object file.

version   Version number of the module.

description   Short description of the module.

srcversion   Hash of the source code used to create the module.

alias   Internal alias names for the module.

depends   Comma-separated list of any modules on which this module depends.

vermagic   Kernel version that was used to compile the module, which is checked against the current
                kernel when the module is loaded.

parm   Module parameters and descriptions.

Modules are loaded into the kernel from kernel object (ko) files in the /lib/
modules/*kernel_version*/kernel directory. To display the absolute path of a kernel object file,
specify the -n option, for example:

```
# modinfo -n parport
/lib/modules/2.6.32-300.27.1.el6uek.x86_64/kernel/drivers/parport/parport.ko
```

For more information, see the lsmod(5) and modinfo(8) manual pages.

6.3 Loading and Unloading Modules

The modprobe command loads kernel modules, for example:

```
# modprobe nfs
# lsmod | grep nfs
nfs                   266415  0
lockd                  66530  1 nfs
fscache                41704  1 nfs
nfs_acl                 2477  1 nfs
auth_rpcgss            38976  1 nfs
sunrpc                204268  5 nfs,lockd,nfs_acl,auth_rpcgss
```

Use the -v verbose option to show if any additional modules are loaded to resolve dependencies.

```
# modprobe -v nfs
insmod /lib/modules/2.6.32-300.27.1.el6uek.x86_64/kernel/net/sunrpc/auth_gss/auth_rpcgss.ko
insmod /lib/modules/2.6.32-300.27.1.el6uek.x86_64/kernel/fs/nfs_common/nfs_acl.ko
insmod /lib/modules/2.6.32-300.27.1.el6uek.x86_64/kernel/fs/fscache/fscache.ko
...
```

To determine the dependencies, the modprobe command queries the /lib/
modules/*kernel_version*/modules.dep file, which the depmod utility creates when you install kernel
modules.

> **Note**
>
> `modprobe` does not reload modules that are already loaded. You must first unload a module before you can load it again.

Use the `-r` option to unload kernel modules, for example:

```
# modprobe -rv nfs
rmmod /lib/modules/2.6.32-300.27.1.el6uek.x86_64/kernel/fs/nfs/nfs.ko
rmmod /lib/modules/2.6.32-300.27.1.el6uek.x86_64/kernel/fs/lockd/lockd.ko
rmmod /lib/modules/2.6.32-300.27.1.el6uek.x86_64/kernel/fs/fscache/fscache.ko
...
```

Modules are unloaded in the reverse order that they were loaded. Modules are not unloaded if a process or another loaded module requires them.

> **Note**
>
> `modprobe` uses the `insmod` and `rmmod` utilities to load and unload modules. As `insmod` and `rmmod` do not resolve module dependencies, do not use these utilities.

For more information, see the `modprobe(8)` and `modules.dep(5)` manual pages.

6.4 About Module Parameters

Modules accept parameters that you can specify using `modprobe` to modify a module's behavior:

```
# modprobe module_name parameter=value ...
```

Use spaces to separate multiple parameter/value pairs. Array values are represented by a comma-separated list, for example:

```
# modprobe foo arrayparm=1,2,3,4
```

You can also change the values of some parameters for loaded modules and built-in drivers by writing the new value to a file under `/sys/module/module_name/parameters`, for example:

```
# echo 0 > /sys/module/ahci/parameters/skip_host_reset
```

The `/etc/modprobe.d` directory contains `.conf` configuration files specify module options, create module aliases, and override the usual behavior of `modprobe` for modules with special requirements. The `/etc/modprobe.conf` file that was used with earlier versions of `modprobe` is also valid if it exists. Entries in the `/etc/modprobe.conf` and `/etc/modprobe.d/*.conf` files use the same syntax.

The following are commonly used commands in `modprobe` configuration files:

`alias`   Creates an alternate name for a module. The alias can include shell wildcards. For example, create an alias for the `sd-mod` module:

```
alias block-major-8-* sd_mod
```

As a result, a command such as `modprobe block-major-8-0` has the same effect as `modprobe sd_mod`.

`blacklist`   Ignore a module's internal alias that is displayed by the `modinfo` command. This command is typically used if the associated hardware is not required, if two or more modules both support the same devices, or if a module invalidly claims to support a device. For example, blacklist the alias for the frame-buffer driver `cirrusfb`:

```
blacklist cirrusfb
```

The `/etc/modprobe.d/blacklist.conf` file prevents hotplug scripts from loading a module, usually so that a different driver binds the module instead, regardless of which driver happens to be probed first.

`install` Runs a shell command instead of loading a module into the kernel. For example, load the module `snd-emu10k1-synth` instead of `snd-emu10k1`:

```
install snd-emu10k1 /sbin/modprobe --ignore-install snd-emu10k1 && \
/sbin/modprobe snd-emu10k1-synth
```

`options` Defines options for a module,. For example, define the `nohwcrypt` and `qos` options for the `b43` module:

```
options b43 nohwcrypt=1 qos=0
```

`remove` Runs a shell command instead of unloading a module. For example, unmount `/proc/fs/nfsd` before unloading the `nfsd` module:

```
remove nfsd { /bin/umount /proc/fs/nfsd > /dev/null 2>&1 || :; } ; \
/sbin/modprobe -r --first-time --ignore-remove nfsd
```

For more information, see the `modprobe.conf(5)` manual page.

6.5 Specifying Modules to be Loaded at Boot Time

The system loads most modules automatically at boot time. If necessary, you can specify an additional module that should be loaded.

To specify a module to be loaded at boot time:

1. Create a file in the `/etc/sysconfig/modules` directory. The file name must have the extension `.modules`, for example `foo.modules`.

2. Edit the file to create the script that loads the module.

   The script to load a module can be a simple `modprobe` call, for example:

   ```
   #!/bin/sh
   modprobe foo
   ```

   or more complex to include error handling:

   ```
   #!/bin/sh
   if [ ! -c /dev/foo ] ; then
     exec /sbin/modprobe foo > /dev/null 2>&1
   fi
   ```

3. Use the following command to make the script executable:

   ```
   # chmod 755 /etc/sysconfig/modules/foo.modules
   ```

Chapter 7 Device Management

Table of Contents

This chapter describes how the system uses device files and how the udev device manager dynamically creates or removes device node files.

7.1 About Device Files

The `/dev` directory contains *device files* (also sometimes known as *device special files* and *device nodes*) that provide access to peripheral devices such as hard disks, to resources on peripheral devices such as disk partitions, and pseudo devices such as a random number generator.

The `/dev` directory has several subdirectory hierarchies, each of which holds device files that relate to a certain type of device. For example, the `/dev/disk/id-by-uuid` directory contains device files for hard disks named according to the universally unique identifier (UUID) for the disk. The device files in subdirectories such as these are actually implemented as symbolic links to device files in `/dev`. You can access the same device using the file in `/dev` or the corresponding link to the file listed in `/dev/disk/id-by-uuid`.

If you use the `ls -l` command to list the files under `/dev`, you see that some device files are shown as being either type `b` for *block* or type `c` for *character*. These devices have a pair of numbers associated with them instead of a file size. These *major* and *minor* numbers identify the device to the system.

```
# ls -l /dev
total 0
crw-rw----. 1 root     root     10,  56 Mar 17 08:17 autofs
drwxr-xr-x. 2 root     root         640 Mar 17 08:17 block
drwxr-xr-x. 2 root     root          80 Mar 17 08:16 bsg
drwxr-xr-x. 3 root     root          60 Mar 17 08:16 bus
lrwxrwxrwx. 1 root     root           3 Mar 17 08:17 cdrom -> sr0
drwxr-xr-x. 2 root     root        2880 Mar 17 08:17 char
crw-------. 1 root     root      5,   1 Mar 17 08:17 console
lrwxrwxrwx. 1 root     root          11 Mar 17 08:17 core -> /proc/kcore
drwxr-xr-x. 4 root     root         100 Mar 17 08:17 cpu
crw-rw----. 1 root     root     10,  61 Mar 17 08:17 cpu_dma_latency
drwxr-xr-x. 6 root     root         120 Mar 17 08:16 disk
brw-rw----. 1 root     disk    253,   0 Mar 17 08:17 dm-0
brw-rw----. 1 root     disk    253,   1 Mar 17 08:17 dm-1
...
crw-rw-rw-. 1 root     root      1,   3 Mar 17 08:17 /dev/null
...
drwxr-xr-x. 2 root      root           0 Mar 17 08:16 pts
...
crw-rw-rw-. 1 root      root      1,   8 Mar 17 08:17 random
...
brw-rw----. 1 root      disk      8,   0 Mar 17 08:17 sda
brw-rw----. 1 root      disk      8,   1 Mar 17 08:17 sda1
brw-rw----. 1 root      disk      8,   2 Mar 17 08:17 sda2
...
lrwxrwxrwx. 1 root      root          15 Mar 17 08:17 stderr -> /proc/self/fd/2
lrwxrwxrwx. 1 root      root          15 Mar 17 08:17 stdin -> /proc/self/fd/0
```

```
lrwxrwxrwx. 1 root     root        15 Mar 17 08:17 stdout -> /proc/self/fd/1
...
crw--w----. 1 root     tty      4,   0 Mar 17 08:17 tty0
crw--w----. 1 root     tty      4,   1 Mar 17 08:17 tty1
...
crw-rw-rw-. 1 root     root     1,   9 Mar 17 08:17 urandom
...
crw-rw-rw-. 1 root     root     1,   5 Mar 17 08:17 zero
```

Block devices support random access to data, seeking media for data, and usually allow data to be buffered while it is being written or read. Examples of block devices include hard disks, CD-ROM drives, flash memory, and other addressable memory devices. The kernel writes data to or reads data from a block device in blocks of a certain number of bytes. In the sample output, `sda` is the block device file that corresponds to the hard disk, and it has a major number of 8 and a minor number of 0. `sda1` and `sda2` are partitions of this disk, and they have the same major number as `sda` (8), but their minor numbers are 1 and 2.

Character devices support streaming of data to or from a device, and data is not usually buffered nor is random access permitted to data on a device. The kernel writes data to or reads data from a character device one byte at a time. Examples of character devices include keyboards, mice, terminals, pseudo-terminals, and tape drives. `tty0` and `tty1` are character device files that correspond to terminal devices that allow users to log in from serial terminals or terminal emulators. These files have major number 4 and minor numbers 0 and 1.

Pseudo-terminals slave devices emulate real terminal devices to interact with software. For example, a user might log in on a terminal device such as `/dev/tty1`, which then uses the pseudo-terminal master device `/dev/pts/ptmx` to interact with an underlying pseudo-terminal device. The character device files for pseudo-terminal slaves and master are located in the `/dev/pts` directory:

```
# ls -l /dev/pts
total 0
crw--w----. 1 guest tty  136, 0 Mar 17 10:11 0
crw--w----. 1 guest tty  136, 1 Mar 17 10:53 1
crw--w----. 1 guest tty  136, 2 Mar 17 10:11 2
c---------. 1 root  root   5, 2 Mar 17 08:16 ptmx
```

Some device entries, such as `stdin` for the standard input, are symbolically linked via the `self` subdirectory of the `proc` file system. The pseudo-terminal device file to which they actually point depends on the context of the process.

```
# ls -l /proc/self/fd/[012]
total 0
lrwx------. 1 root root 64 Mar 17 10:02 0 -> /dev/pts/1
lrwx------. 1 root root 64 Mar 17 10:02 1 -> /dev/pts/1
lrwx------. 1 root root 64 Mar 17 10:02 2 -> /dev/pts/1
```

Character devices such as `null`, `random`, `urandom`, and `zero` are examples of pseudo-devices that provide access to virtual functionality implemented in software rather than to physical hardware.

`/dev/null` is a data sink. Data that you write to `/dev/null` effectively disappears but the write operation succeeds. Reading from `/dev/null` returns `EOF` (end-of-file).

`/dev/zero` is a data source of an unlimited number of zero-value bytes.

`/dev/random` and `/dev/urandom` are data sources of streams of pseudo-random bytes. To maintain high-entropy output, `/dev/random` blocks if its entropy pool does not contains sufficient bits of noise. `/dev/urandom` does not block and, as a result, the entropy of its output might not be as consistently high as that of `/dev/random`. However, neither `/dev/random` nor `/dev/urandom` are considered to be truly random enough for the purposes of secure cryptography such as military-grade encryption.

You can find out the size of the entropy pool and the entropy value for `/dev/random` from virtual files under `/proc/sys/kernel/random`:

```
# cat /proc/sys/kernel/random/poolsize
4096
# cat /proc/sys/kernel/random/entropy_avail
3467
```

For more information, see the `null(4)`, `pts(4)`, and `random(4)` manual pages.

7.2 About the Udev Device Manager

The udev device manager dynamically creates or removes device node files at boot time or if you add a device to or remove a device from the system with a 2.6 version kernel or later. When creating a device node, udev reads the device's `/sys` directory for attributes such as the label, serial number, and bus device number.

Udev can use persistent device names to guarantee consistent naming of devices across reboots, regardless of their order of discovery. Persistent device names are especially important when using external storage devices.

The configuration file for udev is `/etc/udev/udev.conf`, in which you can define the following variables:

`udev_log`   The logging priority, which can be set to `err`, `info` and `debug`. The default value is `err`.

`udev_root`   Specifies the location of the device nodes. The default value is `/dev`.

For more information, see the `udev(7)` manual page.

7.3 About Udev Rules

Udev uses rules files that determine how it identifies devices and creates device names. The udev service (`systemd-udevd`) reads the rules files at system startup and stores the rules in memory. If the kernel discovers a new device or an existing device goes offline, the kernel sends an event action (*uevent*) notification to udev, which matches the in-memory rules against the device attributes in `/sys` to identify the device. As part of device event handling, rules can specify additional programs that should run to configure a device. Rules files, which have the file extension `.rules`, are located in the following directories:

`/lib/udev/rules.d`   Contains default rules files. Do not edit these files.

`/etc/udev/rules.d/`   Contains customized rules files. You can modify these files.
`*.rules`

`/dev/.udev/rules.d/`   Contains temporary rules files. Do not edit these files.
`*.rules`

Udev processes the rules files in lexical order, regardless of which directory they are located. Rules files in `/etc/udev/rules.d` override files of the same name in `/lib/udev/rules.d`.

The following rules are extracted from the file `/lib/udev/rules.d/50-udev- default.rules` and illustrate the syntax of udev rules.

```
# do not edit this file, it will be overwritten on update

SUBSYSTEM=="block", SYMLINK{unique}+="block/%M:%m"
SUBSYSTEM!="block", SYMLINK{unique}+="char/%M:%m"

KERNEL=="pty[pqrstuvwxyzabcdef][0123456789abcdef]", GROUP="tty", MODE="0660"
```

```
KERNEL=="tty[pqrstuvwxyzabcdef][0123456789abcdef]", GROUP="tty", MODE="0660"
...
# mem
KERNEL=="null|zero|full|random|urandom", MODE="0666"
KERNEL=="mem|kmem|port|nvram",  GROUP="kmem", MODE="0640"
...
# block
SUBSYSTEM=="block", GROUP="disk"

...
# network
KERNEL=="tun",                   MODE="0666"
KERNEL=="rfkill",                MODE="0644"

# CPU
KERNEL=="cpu[0-9]*",            MODE="0444"
...
# do not delete static device nodes
ACTION=="remove", NAME=="", TEST=="/lib/udev/devices/%k", \
    OPTIONS+="ignore_remove"
ACTION=="remove", NAME=="?*", TEST=="/lib/udev/devices/$name", \
    OPTIONS+="ignore_remove"
```

Comment lines begin with a `#` character. All other non-blank lines define a rule, which is a list of one or more comma-separated key-value pairs. A rule either assigns a value to a key or it tries to find a match for a key by comparing its current value with the specified value. The following table shows the assignment and comparison operators that you can use.

Operator	Description
=	Assign a value to a key, overwriting any previous value.
+=	Assign a value by appending it to the key's current list of values.
:=	Assign a value to a key. This value cannot be changed by any further rules.
==	Match the key's current value against the specified value for equality.
!=	Match the key's current value against the specified value for equality.

You can use the following shell-style pattern matching characters in values.

Character	Description
?	Matches a single character.
*	Matches any number of characters, including zero.
[]	Matches any single character or character from a range of characters specified within the brackets. For example, `tty[sS][0-9]` would match `ttys7` or `ttyS7`.

The following table lists commonly used match keys in rules.

Match Key	Description
ACTION	Matches the name of the action that led to an event. For example, `ACTION="add"` or `ACTION="remove"`.
ENV{key}	Matches a value for the device property `key`. For example, `ENV{DEVTYPE}=="disk"`.
KERNEL	Matches the name of the device that is affected by an event. For example, `KERNEL=="dm-*"` for disk media.
NAME	Matches the name of a device file or network interface. For example, `NAME="?*"` for any name that consists of one or more characters.

Match Key	Description
SUBSYSTEM	Matches the subsystem of the device that is affected by an event. For example, `SUBSYSTEM=="tty"`.
TEST	Tests if the specified file or path exists. For example, `TEST=="/lib/udev/devices/$name"`, where `$name` is the name of the currently matched device file.

Other match keys include `ATTR{filename}`, `ATTRS{filename}`, `DEVPATH`, `DRIVER`, `DRIVERS`, `KERNELS`, `PROGRAM`, `RESULT`, `SUBSYSTEMS`, and `SYMLINK`.

The following table lists commonly used assignment keys in rules.

Assignment Key	Description	
ENV{key}	Specifies a value for the device property `key`. For example, `GROUP="disk"`.	
GROUP	Specifies the group for a device file. For example, `GROUP="disk"`.	
IMPORT{type}	Specifies a set of variables for the device property, depending on `type`:	
	cmdline	Import a single property from the boot `kernel` command line. For simple flags, udev sets the value of the property to 1. For example, `IMPORT{cmdline}="nodmraid"`.
	db	Interpret the specified value as an index into the device database and import a single property, which must have already been set by an earlier event. For example, `IMPORT{db}="DM_UDEV_LOW_PRIORITY_FLAG"`.
	file	Interpret the specified value as the name of a text file and import its contents, which must be in environmental key format. For example, `IMPORT{file}="keyfile"`.
	parent	Interpret the specified value as a key-name filter and import the stored keys from the database entry for the parent device. For example `IMPORT{parent}="ID_*"`.
	program	Run the specified value as an external program and imports its result, which must be in environmental key format. For example `IMPORT{program}="usb_id --export %p"`.
MODE	Specifies the permissions for a device file. For example, `MODE="0640"`.	
NAME	Specifies the name of a device file. For example, `NAME="em1"`.	
OPTIONS	Specifies rule and device options. For example, `OPTIONS+="ignore_remove"`, which means that the device file is not removed if the device is removed.	
OWNER	Specifies the owner for a device file. For example, `GROUP="root"`.	
RUN	Specifies a command to be run after the device file has been created. For example, `RUN+="/usr/bin/eject $kernel"`, where `$kernel` is the kernel name of the device.	
SYMLINK	Specifies the name of a symbolic link to a device file. For example, `SYMLINK+="disk/by-uuid/$env{ID_FS_UUID_ENC}"`, where `$env{}` is substituted with the specified device property.	

Other assignment keys include `ATTR{key}`, `GOTO`, `LABEL`, `RUN`, and `WAIT_FOR`.

The following table shows string substitutions that are commonly used with the `GROUP`, `MODE`, `NAME`, `OWNER`, `PROGRAM`, `RUN`, and `SYMLINK` keys.

String Substitution	Description
`$attr{file}` or `%s{file}`	Specifies the value of a device attribute from a file under `/sys`. For example, `ENV{MATCHADDR}="$attr{address}"`.
`$devpath` or `%p`	The device path of the device in the `sysfs` file system under `/sys`. For example, `RUN+="keyboard-force-release.sh $devpath common-volume-keys"`.
`$env{key}` or `%E{key}`	Specifies the value of a device property. For example, `SYMLINK+="disk/by-id/` `md-name-$env{MD_NAME}-part%n"`.
`$kernel` or `%k`	The kernel name for the device.
`$major` or `%M`	Specifies the major number of a device. For example, `IMPORT{program}="udisks-dm-export %M %m"`.
`$minor` or `%m`	Specifies the minor number of a device. For example, `RUN` `+="$env{LVM_SBIN_PATH}/lvm pvscan --cache --major $major --` `minor $minor"`.
`$name`	Specifies the device file of the current device. For example, `TEST=="/lib/udev/` `devices/$name"`.

Udev expands the strings specified for `RUN` immediately before its program is executed, which is after udev has finished processing all other rules for the device. For the other keys, udev expands the strings while it is processing the rules.

For more information, see the `udev(7)` manual page.

7.4 Querying Udev and Sysfs

You can use the `udevadm` command to query the udev database and `sysfs`.

For example, to query the `sysfs` device path relative to `/sys` that corresponds to the device file `/dev/sda`:

```
# udevadm info --query=path --name=/dev/sda
/devices/pci0000:00/0000:00:0d.0/host0/target0:0:0/0:0:0:0/block/sda
```

To query the symbolic links that point to `/dev/sda`:

```
# udevadm info --query=symlink --name=/dev/sda
block/8:0
disk/by-id/ata-VBOX_HARDDISK_VB6ad0115d-356e4c09
disk/by-id/scsi-SATA_VBOX_HARDDISK_VB6ad0115d-356e4c09
disk/by-path/pci-0000:00:0d.0-scsi-0:0:0:0
```

The paths are relative to `udev_root` (by default, `/dev`).

To query the properties of `/dev/sda`:

```
# udevadm info --query=property --name=/dev/sda
UDEV_LOG=3
DEVPATH=/devices/pci0000:00/0000:00:0d.0/host0/target0:0:0/0:0:0:0/block/sda
MAJOR=8
MINOR=0
```

```
DEVNAME=/dev/sda
DEVTYPE=disk
SUBSYSTEM=block
ID_ATA=1
ID_TYPE=disk
ID_BUS=ata
ID_MODEL=VBOX_HARDDISK
ID_MODEL_ENC=VBOX\x20HARDDISK\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20...
ID_REVISION=1.0
ID_SERIAL=VBOX_HARDDISK_VB579a85b0-bf6debae
ID_SERIAL_SHORT=VB579a85b0-bf6debae
ID_ATA_WRITE_CACHE=1
ID_ATA_WRITE_CACHE_ENABLED=1
ID_ATA_FEATURE_SET_PM=1
ID_ATA_FEATURE_SET_PM_ENABLED=1
ID_ATA_SATA=1
ID_ATA_SATA_SIGNAL_RATE_GEN2=1
ID_SCSI_COMPAT=SATA_VBOX_HARDDISK_VB579a85b0-bf6debae
ID_PATH=pci-0000:00:0d.0-scsi-0:0:0:0
ID_PART_TABLE_TYPE=dos
LVM_SBIN_PATH=/sbin
UDISKS_PRESENTATION_NOPOLICY=0
UDISKS_PARTITION_TABLE=1
UDISKS_PARTITION_TABLE_SCHEME=mbr
UDISKS_PARTITION_TABLE_COUNT=2
UDISKS_ATA_SMART_IS_AVAILABLE=0
DEVLINKS=/dev/block/8:0 /dev/disk/by-id/ata-VBOX_HARDDISK_VB579a85b0-bf6debae ...
```

To query all information for `/dev/sda`:

```
# udevadm info --query=all --name=/dev/sda
P: /devices/pci0000:00/0000:00:0d.0/host0/target0:0:0/0:0:0:0/block/sda
N: sda
W: 37
S: block/8:0
S: disk/by-id/ata-VBOX_HARDDISK_VB579a85b0-bf6debae
S: disk/by-id/scsi-SATA_VBOX_HARDDISK_VB579a85b0-bf6debae
S: disk/by-path/pci-0000:00:0d.0-scsi-0:0:0:0
E: UDEV_LOG=3
E: DEVPATH=/devices/pci0000:00/0000:00:0d.0/host0/target0:0:0/0:0:0:0/block/sda
E: MAJOR=8
E: MINOR=0
E: DEVNAME=/dev/sda
E: DEVTYPE=disk
E: SUBSYSTEM=block
E: ID_ATA=1
E: ID_TYPE=disk
E: ID_BUS=ata
E: ID_MODEL=VBOX_HARDDISK
E: ID_MODEL_ENC=VBOX\x20HARDDISK\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20...
E: ID_SERIAL=VBOX_HARDDISK_VB579a85b0-bf6debae
E: ID_SERIAL_SHORT=VB579a85b0-bf6debae
E: ID_ATA_WRITE_CACHE=1
E: ID_ATA_WRITE_CACHE_ENABLED=1
E: ID_ATA_FEATURE_SET_PM=1
E: ID_ATA_FEATURE_SET_PM_ENABLED=1
E: ID_ATA_SATA=1
E: ID_ATA_SATA_SIGNAL_RATE_GEN2=1
E: ID_SCSI_COMPAT=SATA_VBOX_HARDDISK_VB579a85b0-bf6debae
E: ID_PATH=pci-0000:00:0d.0-scsi-0:0:0:0
E: ID_PART_TABLE_TYPE=dos
E: LVM_SBIN_PATH=/sbin
E: UDISKS_PRESENTATION_NOPOLICY=0
E: UDISKS_PARTITION_TABLE=1
E: UDISKS_PARTITION_TABLE_SCHEME=mbr
E: UDISKS_PARTITION_TABLE_COUNT=2
E: UDISKS_ATA_SMART_IS_AVAILABLE=0
```

```
E: DEVLINKS=/dev/block/8:0 /dev/disk/by-id/ata-VBOX_HARDDISK_VB579a85b0-bf6debae ...
```

To display all properties of `/dev/sda` and its parent devices that udev has found in `/sys`:

```
# udevadm info --attribute-walk --name=/dev/sda
...
  looking at device '/devices/pci0000:00/0000:00:0d.0/host0/target0:0:0/0:0:0:0/block/sda':
    KERNEL=="sda"
    SUBSYSTEM=="block"
    DRIVER==""
    ATTR{range}=="16"
    ATTR{ext_range}=="256"
    ATTR{removable}=="0"
    ATTR{ro}=="0"
    ATTR{size}=="83886080"
    ATTR{alignment_offset}=="0"
    ATTR{capability}=="52"
    ATTR{stat}=="   20884    15437  1254282   338919     5743     8644   103994   109005 ...
    ATTR{inflight}=="        0        0"

  looking at parent device '/devices/pci0000:00/0000:00:0d.0/host0/target0:0:0/0:0:0:0':
    KERNELS=="0:0:0:0"
    SUBSYSTEMS=="scsi"
    DRIVERS=="sd"
    ATTRS{device_blocked}=="0"
    ATTRS{type}=="0"
    ATTRS{scsi_level}=="6"
    ATTRS{vendor}=="ATA     "
    ATTRS{model}=="VBOX HARDDISK   "
    ATTRS{rev}=="1.0 "
    ATTRS{state}=="running"
    ATTRS{timeout}=="30"
    ATTRS{iocounterbits}=="32"
    ATTRS{iorequest_cnt}=="0x6830"
    ATTRS{iodone_cnt}=="0x6826"
    ATTRS{ioerr_cnt}=="0x3"
    ATTRS{modalias}=="scsi:t-0x00"
    ATTRS{evt_media_change}=="0"
    ATTRS{dh_state}=="detached"
    ATTRS{queue_depth}=="31"
    ATTRS{queue_ramp_up_period}=="120000"
    ATTRS{queue_type}=="simple"

  looking at parent device '/devices/pci0000:00/0000:00:0d.0/host0/target0:0:0':
    KERNELS=="target0:0:0"
    SUBSYSTEMS=="scsi"
    DRIVERS==""

  looking at parent device '/devices/pci0000:00/0000:00:0d.0/host0':
    KERNELS=="host0"
    SUBSYSTEMS=="scsi"
    DRIVERS==""

  looking at parent device '/devices/pci0000:00/0000:00:0d.0':
    KERNELS=="0000:00:0d.0"
    SUBSYSTEMS=="pci"
    DRIVERS=="ahci"
    ATTRS{vendor}=="0x8086"
    ATTRS{device}=="0x2829"
    ATTRS{subsystem_vendor}=="0x0000"
    ATTRS{subsystem_device}=="0x0000"
    ATTRS{class}=="0x010601"
    ATTRS{irq}=="21"
    ATTRS{local_cpus}=="00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000003"
    ATTRS{local_cpulist}=="0-1"
    ATTRS{modalias}=="pci:v00008086d00002829sv00000000sd00000000bc01sc06i01"
    ATTRS{numa_node}=="-1"
```

```
    ATTRS{enable}=="1"
    ATTRS{broken_parity_status}=="0"
    ATTRS{msi_bus}==""
    ATTRS{msi_irqs}==""

  looking at parent device '/devices/pci0000:00':
    KERNELS=="pci0000:00"
    SUBSYSTEMS==""
    DRIVERS==""
```

The command starts at the device specified by its device path and walks up the chain of parent devices. For every device that it finds, it displays all possible attributes for the device and its parent devices in the match key format for udev rules.

For more information, see the udevadm(8) manual page.

7.5 Modifying Udev Rules

The order in which rules are evaluated is important. Udev processes rules in lexical order. If you want to add your own rules, you need udev to find and evaluate these rules before the default rules.

The following example illustrates how to implement a udev rules file that adds a symbolic link to the disk device /dev/sdb.

1. Create a rule file under /etc/udev/rules.d with a file name such as 10-local.rules that udev will read before any other rules file.

   For example, the following rule in 10-local.rules creates the symbolic link /dev/my_disk, which points to /dev/sdb:

   ```
   KERNEL=="sdb", ACTION=="add", SYMLINK="my_disk"
   ```

   Listing the device files in /dev shows that udev has not yet applied the rule:

   ```
   # ls /dev/sd* /dev/my_disk
   ls: cannot access /dev/my_disk: No such file or directory
   /dev/sda   /dev/sda1   /dev/sda2   /dev/sdb
   ```

2. To simulate how udev applies its rules to create a device, you can use the udevadm test command with the device path of sdb listed under the /sys/class/block hierarchy, for example:

   ```
   # udevadm test /sys/class/block/sdb
   calling: test
   version ...
   This program is for debugging only, it does not run any program
   specified by a RUN key. It may show incorrect results, because
   some values may be different, or not available at a simulation run.
   ...
   LINK 'my_disk' /etc/udev/rules.d/10-local.rules:1
   ...
   creating link '/dev/my_disk' to '/dev/sdb'
   creating symlink '/dev/my_disk' to 'sdb
   ...
   ACTION=add
   DEVLINKS=/dev/disk/by-id/ata-VBOX_HARDDISK_VB186e4ce2-f80f170d
     /dev/disk/by-uuid/a7dc508d-5bcc-4112-b96e-f40b19e369fe
     /dev/my_disk
   ...
   ```

3. Restart the systemd-udevd service:

   ```
   # systemctl restart systemd-udevd
   ```

After udev processes the rules files, the symbolic link `/dev/my_disk` has been added:

```
# ls -F /dev/sd* /dev/my_disk
/dev/my_disk@  /dev/sda  /dev/sda1  /dev/sda2  /dev/sdb
```

To undo the changes, remove `/etc/udev/rules.d/10-local.rules` and `/dev/my_disk` and run `systemctl restart systemd-udevd` again.

Chapter 8 Task Management

Table of Contents

This chapter describes how to configure the system to run tasks automatically within a specific period of time, at a specified time and date, or when the system is lightly loaded.

8.1 About Automating Tasks

You can use automated tasks to perform periodic backups, monitor the system, run custom scripts, and other administrative tasks.

The `cron` and `anacron` utilities allow you to schedule the execution of recurring tasks (*jobs*) according to a combination of the time, day of the month, month, day of the week, and week. `cron` allows you to schedule jobs to run as often as every minute. If the system is down when a job is scheduled, `cron` does not run the job when the system restarts. `anacron` allows you to schedule a system job to run only once per day. However, if a scheduled job has not been run, that job runs when the system restarts. `anacron` is mainly intended for use on laptop computers.

You do not usually need to run `cron` and `anacron` directly. The `crond` daemon executes scheduled tasks on behalf of `cron` and it starts `anacron` once every hour. `crond` looks in `/etc/crontab` or in files in `/etc/cron.d` for system `cron` job definitions, and `/var/spool/cron` for `cron` job definitions belonging to users. `crond` checks each job definition to see whether it should run in the current minute. If a job is scheduled for execution, `crond` runs it as the owner of the job definition file or, for system `cron` jobs, the user specified in the job definition (if any).

`crond` runs the `0anacron` script in the `/etc/cron.hourly` directory as `root` once per hour according to the schedule in `/etc/cron.d/0hourly`. If `anacron` is not already running and the system is connected to mains and not battery power, `crond` starts `anacron`.

`anacron` runs the scripts in the `/etc/cron.daily`, `/etc/cron.weekly`, and `/etc/cron.monthly` directories as `root` once per day, week or month, according to the job definitions that are scheduled in `/etc/anacrontab`.

8.2 Configuring cron Jobs

System `cron` jobs are defined in `crontab`-format files in `/etc/crontab` or in files in `/etc/cron.d`. A `crontab` file usually consists of definitions for the `SHELL`, `PATH`, `MAILTO`, and `HOME` variables for the environment in which the jobs run, followed by the job definitions themselves. Comment lines start with a `#` character. Job definitions are specified in the following format:

```
minute  hour  day  month  day-of-week  user  command
```

where the fields are:

minute   0-59.

`hour`	0-23.
`day`	1-31.
`month`	1-12 or `jan`, `feb`,..., `dec`.
`day-of-week`	0-7 (Sunday is 0 or 7) or `sun`, `mon`,...,`sat`.
`user`	The user to run the command as, or `*` for the owner of the `crontab` file.
`command`	The shell script or command to be run.

For the `minute` through `day-of week` fields, you can use the following special characters:

* `*`   (asterisk) All valid values for the field.

* `-`   (dash) A range of integers, for example, `1-5`.

* `,`   (comma) A list of values, for example, `0,2,4`.

* `/`   (forward slash) A step value, for example, `/3` in the `hour` field means every three hours.

For example, the following entry would run a command every five minutes on weekdays:

```
0-59/5  *  *  *  1-5  *   command
```

Run a command at one minute past midnight on the first day of the months April, June, September, and November:

```
1  0  1  4,6,9,11  *  *   command
```

`root` can add job definition entries to `/etc/crontab`, or add `crontab`-format files to the `/etc/cron.d` directory.

> **Note**
>
> If you add an executable job script to the `/etc/cron.hourly` directory, `crond` runs the script once every hour. Your script should check that it is not already running.

For more information, see the `crontab(5)` manual page.

8.2.1 Controlling Access to Running cron Jobs

If permitted, users other than `root` can configure `cron` tasks by using the `crontab` utility. All user-defined `crontab`-format files are stored in the `/var/spool/cron` directory with the same name as the users that created them.

`root` can use the `/etc/cron.allow` and `/etc/cron.deny` files to restrict access to `cron`. `crontab` checks the access control files each time that a user tries to add or delete a `cron` job. If `/etc/cron.allow` exists, only users listed in it are allowed to use `cron`, and `/etc/cron.deny` is ignored. If `/etc/cron.allow` does not exist, users listed in `/etc/cron.deny` are not allowed to use `cron`. If neither file exists, only `root` can use `cron`. The format of both `/etc/cron.allow` and `/etc/cron.deny` is one user name on each line.

To create or edit a `crontab` file as a user, log in as that user and type the command `crontab -e`, which opens your `crontab` file in the `vi` editor (or the editor specified by the `EDITOR` or `VISUAL` environment

variables). The file has the same format as `/etc/crontab` except that the user field is omitted. When you save changes to the file, these are written to the file `/var/spool/cron/`*`username`*. To list the contents of your `crontab` file, use the `crontab -l` command. To delete your `crontab` file, use the `crontab -r` command.

For more information, see the `crontab(1)` manual page.

8.3 Configuring anacron Jobs

System `anacron` jobs are defined in `/etc/anacrontab`, which contains definitions for the `SHELL`, `PATH`, `MAILTO`, `RANDOM_DELAY`, and `START_HOURS_RANGE` variables for the environment in which the jobs run, followed by the job definitions themselves. Comment lines start with a `#` character.

`RANDOM_DELAY` is the maximum number of random time in minutes that `anacron` adds to the *`delay`* parameter for a job. The default minimum delay is 6 minutes. The random offset is intended to prevent `anacron` overloading the system with too many jobs at the same time.

`START_HOURS_RANGE` is the time range of hours during the day when anacron can run scheduled jobs.

Job definitions are specified in the following format:

```
period   delay   job-id   command
```

where the fields are:

period Frequency of job execution specified in days or as `@daily`, `@weekly`, or `@monthly` for once per day, week, or month.

delay Number of minutes to wait before running a job.

job-id Unique name for the job in log files.

command The shell script or command to be run.

The following entries are taken from the default `/etc/anacrontab` file:

```
SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
# the maximal random delay added to the base delay of the jobs
RANDOM_DELAY=45
# the jobs will be started during the following hours only
START_HOURS_RANGE=3-22

#period in days  delay in minutes  job-identifier   command
1                5                 cron.daily       nice run-parts /etc/cron.daily
7                25                cron.weekly      nice run-parts /etc/cron.weekly
@monthly         45                cron.monthly     nice run-parts /etc/cron.monthly
```

By default, `anacron` runs jobs between 03:00 and 22:00 and randomly delays jobs by between 11 and 50 minutes. The job scripts in `/etc/cron.daily`, run anywhere between 03:11 and 03:50 every day if the system is running, or after the system is booted and the time is less than 22:00. The `run-parts` script sequentially executes every program within the directory specified as its argument.

Scripts in `/etc/cron.weekly` run once per week with a delay offset of between 31 and 70 minutes.

Scripts in `/etc/cron.monthly` run once per week with a delay offset of between 51 and 90 minutes.

For more information, see the `anacron(8)` and `anacrontab(5)` manual pages.

8.4 Running One-time Tasks

You can use the `at` command to schedule a one-time task to run at a specified time, or the `batch` command to schedule a one-time task to run when the system load average drops below 0.8. The `atd` service must be running to use `at` or `batch`.

```
# systemctl is-active atd
active
```

`at` takes a time as its argument and reads the commands to be run from the standard input. For example, run the commands in the file `atjob` in 20 minutes time:

```
# at now + 20 minutes < ./atjob
job 1 at 2013-03-19 11:25
```

The `atq` command shows the `at` jobs that are queued to run:

```
# atq
1 2013-03-19 11:25 a root
```

The `batch` command also reads command from the standard input, but it does not run until the system load average drops below 0.8. For example:

```
# batch < batchjob
job 2 at 2013-03-19 11:31
```

To cancel one or more queued jobs, specify their job numbers to the `atrm` command, for example:

```
# atrm 1 2
```

For more information, see the `at(1)` manual page.

8.4.1 Changing the Behavior of Batch Jobs

The load average of a system, as displayed by the `uptime` and `w` commands, represents the average number of processes that are queued to run on the CPUs or CPU cores over a given time period. Typically, a system might not considered overloaded until the load average exceeds 0.8 times the number of CPUs or CPU cores. On such systems, you would usually want `atd` to be able to run batch jobs when the load average drops below the number of CPUs or CPU cores, rather than the default limit of 0.8. For example, on a system with 4 CPU cores, you could set the load-average limit above which `atd` will not run batch jobs to 3.2.

If you know that a batch job typically takes more than a minute to run, you can also change the minimum interval that `atd` waits between starting batch jobs. The default minimum interval is 60 seconds.

To change the load-average limit and minimum interval time for batch jobs:

1. Edit the `atd` configuration file, `/etc/sysconfig/atd`, uncomment the line that defines the `OPTS` variable, and edit the line to specify the new load-average limit and minimum interval time, for example:

    ```
    OPTS="-b 100 -l 3"
    ```

    This example sets the minimum interval to 100 seconds and the load-average limit to 3.

2. Restart the `atd` service:

    ```
    # systemctl restart atd
    ```

3. Verify that the `atd` daemon is running with the new minimum interval and load-average limit:

```
# systemctl status atd
atd.service - Job spooling tools
   Loaded: loaded (/usr/lib/systemd/system/atd.service; enabled)
   Active: active (running) since Mon 2014-04-28 15:37:04 BST; 2min 53s ago
 Main PID: 6731 (atd)
   CGroup: /system.slice/atd.service
           └─6731 /usr/sbin/atd -f -b 100 -l 3

Apr 28 15:37:04 localhost.localdomain systemd[1]: Started Job spooling tools.
```

For more information, see the `systemctl(1)` and `atd(8)` manual pages.

Chapter 9 System Monitoring and Tuning

Table of Contents

This chapter describes how to collect diagnostic information about a system for Oracle Support, and how to monitor and tune the performance of a system.

9.1 About sosreport

The `sosreport` utility collects information about a system such as hardware configuration, software configuration, and operational state. You can also use `sosreport` to enable diagnostics and analytical functions. To assist in troubleshooting a problem, `sosreport` records the information in a compressed file that you can send to a support representative.

9.1.1 Configuring and Using sosreport

If the `sos` package is not already installed on your system, use `yum` to install it.

Use the following command to list the available plugins and plugin options.

```
# sosreport -l
The following plugins are currently enabled:

acpid             acpid related information
anaconda          Anaconda / Installation information
.
.
.
The following plugins are currently disabled:

amd               Amd automounter information
cluster           cluster suite and GFS related information
.
.
.
The following plugin options are available:
apache.log             off gathers all apache logs
auditd.syslogsize      15 max size (MiB) to collect per syslog file
.
.
.
```

See the `sosreport(1)` manual page for information about how to enable or disable plugins, and how to set values for plugin options.

To run `sosreport`:

1. Enter the command, specifying any options that you need to tailor the report to report information about a problem area.

```
# sosreport [options ...]
```

For example, to record only information about Apache and Tomcat, and to gather all the Apache logs:

```
# sosreport -o apache,tomcat -k apache.log=on

sosreport (version 2.2)
.
.
.
Press ENTER to continue, or CTRL-C to quit.
```

To enable all boolean options for all loaded plugins except the `rpm.rpmva` plugin that verifies all packages, and which takes a considerable time to run:

```
# sosreport -a -k rpm.rpmva=off
```

2. Type Enter, and enter additional information when prompted.

```
Please enter your first initial and last name [email_address]: AName
Please enter the case number that you are generating this report for: case#

  Running plugins. Please wait ...

  Completed [55/55] ...
Creating compressed archive...

Your sosreport has been generated and saved in:
  /tmp/sosreport-AName.case#-datestamp-ID.tar.xz

The md5sum is: checksum

Please send this file to your support representative.
```

`sosreport` saves the report as an `xz`-compressed `tar` file in `/tmp`.

For more information, see the `sosreport(1)` manual page.

9.2 About System Performance Tuning

Performance issues can be caused by any of a system's components, software or hardware, and by their interaction. Many performance diagnostics utilities are available for Oracle Linux, including tools that monitor and analyze resource usage by different hardware components and tracing tools for diagnosing performance issues in multiple processes and their threads.

9.2.1 About Performance Problems

Many performance issues are the result of configuration errors. You can avoid such errors by using a validated configuration that has been pre-tested fore the supported software, hardware, storage, drivers, and networking components. A validated configuration incorporates the best practices for Oracle Linux deployment and has undergone real-world testing of the complete stack. Oracle publishes more than 100 validated configurations, which are freely available for download. You should also refer to the release notes for recommendations on setting kernel parameters.

A typical problem involves out of memory errors and generally poor performance when running Oracle Database. The cause of this problem is likely to be that the system is not configured to use the HugePages feature for the System Global Area (SGA). With HugePages, you can set the page size to between 2MB and 256MB, so reducing the total number of pages that the kernel needs to manage. The memory associated with HugePages cannot be swapped out, which forces the SGA to remain resident in memory.

The following utilities allow you to collect information about system resource usage and errors, and can help you to identify performance problems caused by overloaded disks, network, memory, or CPUs:

`dmesg`   Displays the contents of the kernel ring buffer, which can contain errors about system resource usage. Provided by the `util-linux-ng` package.

`dstat`   Displays statistics about system resource usage. Provided by the `dstat` package.

`free`   Displays the amount of free and used memory in the system. Provided by the `procps` package.

`iostat`   Reports I/O statistics. Provided by the `sysstat` package.

`iotop`   Monitors disk and swap I/O on a per-process basis. Provided by the `iotop` package.

`ip`   Reports network interface statistics and errors. Provided by the `iproute` package.

`mpstat`   Reports processor-related statistics. Provided by the `sysstat` package.

`sar`   Reports information about system activity. Provided by the `sysstat` package.

`ss`   Reports network interface statistics. Provided by the `iproute` package.

`top`   Provides a dynamic real-time view of the tasks that are running on a system. Provided by the `procps` package.

`uptime`   Displays the system load averages for the past 1, 5, and 15 minutes. Provided by the `procps` package.

`vmstat`   Reports virtual memory statistics. Provided by the `procps` package.

Many of these utilities provide overlapping functionality. For more information, see the individual manual page for the utility.

See Section 5.2.3, "Parameters that Control System Performance" for a list of kernel parameters that affect system performance.

9.2.2 Monitoring Usage of System Resources

You need to collect and monitor system resources regularly to provide you with a continuous record of a system. Establish a baseline of acceptable measurements under typical operating conditions. You can then use the baseline as a reference point to make it easier to identify memory shortages, spikes in resource usage, and other problems when they occur. Monitoring system performance also allows you to plan for future growth and to see how configuration changes might affect future performance.

To run a monitoring command every *interval* seconds in real time and watch its output change, use the watch command. For example, the following command runs the `mpstat` command once per second:

```
# watch -n interval mpstat
```

Alternatively, many of the commands allow you to specify the sampling interval in seconds, for example:

```
# mpstat interval
```

If installed, the `sar` command records statistics every 10 minutes while the system is running and retains this information for every day of the current month. The following command displays all the statistics that `sar` recorded for day *DD* of the current month:

```
# sar -A -f /var/log/sa/saDD
```

To run `sar` command as a background process and collect data in a file that you can display later by using the `-f` option:

```
# sar -o datafile interval count >/dev/null 2>&1 &
```

where `count` is the number of samples to record.

Oracle OSWatcher Black Box (OSWbb) and OSWbb analyzer (OSWbba) are useful tools for collecting and analysing performance statistics. For more information, see Section 9.2.4, "About OSWatcher Black Box".

9.2.2.1 Monitoring CPU Usage

The `uptime`, `mpstat`, `sar`, `dstat`, and `top` utilities allow you to monitor CPU usage. When a system's CPU cores are all occupied executing the code of processes, other processes must wait until a CPU core becomes free or the scheduler switches a CPU core to run their code. If too many processes are queued too often, this can represent a bottleneck in the performance of the system.

The commands `mpstat -P ALL` and `sar -u -P ALL` display CPU usage statistics for each CPU core and averaged across all CPU cores.

The `%idle` value shows the percentage of time that a CPU was not running system code or process code. If the value of `%idle` is near 0% most of the time on all CPU cores, the system is CPU-bound for the workload that it is running. The percentage of time spent running system code (`%system` or `%sys`) should not usually exceed 30%, especially if `%idle` is close to 0%.

The system load average represents the number of processes that are running on CPU cores, waiting to run, or waiting for disk I/O activity to complete averaged over a period of time. On a busy system, the load average reported by `uptime` or `sar -q` should usually be not greater than two times the number of CPU cores over periods as long as 5 or 15 minutes. If the load average exceeds four times the number of CPU cores for long periods, the system is overloaded.

In addition to load averages (`ldavg-*`), the `sar -q` command reports the number of processes currently waiting to run (the *run-queue size*, `runq-sz`) and the total number of processes (`plist_sz`). The value of `runq-sz` also provides an indication of CPU saturation.

Determine the system's average load under normal loads where users and applications do not experience problems with system responsiveness, and then look for deviations from this benchmark over time. A dramatic rise in the load average can indicate a serious performance problem.

A combination of sustained large load average or large run queue size and low `%idle` can indicate that the system has insufficient CPU capacity for the workload. When CPU usage is high, use a command such as `dstat` or `top` to determine which processes are most likely to be responsible. For example, the following `dstat` command shows which processes are using CPUs, memory, and block I/O most intensively:

```
# dstat --top-cpu --top-mem --top-bio
```

The `top` command provides a real-time display of CPU activity. By default, `top` lists the most CPU-intensive processes on the system. In its upper section, `top` displays general information including the load averages over the past 1, 5 and 15 minutes, the number of running and sleeping processes (tasks), and total CPU and memory usage. In its lower section, `top` displays a list of processes, including the process ID number (PID), the process owner, CPU usage, memory usage, running time, and the command name. By default, the list is sorted by CPU usage, with the top consumer of CPU listed first. Type `f` to select which fields `top` displays, `o` to change the order of the fields, or `O` to change the sort field. For example, entering **On** sorts the list on the percentage memory usage field (`%MEM`).

9.2.2.2 Monitoring Memory Usage

The `sar -r` command reports memory utilization statistics, including `%memused`, which is the percentage of physical memory in use.

`sar -B` reports memory paging statistics, including `pgscank/s`, which is the number of memory pages scanned by the `kswapd` daemon per second, and `pgscand/s`, which is the number of memory pages scanned directly per second.

`sar -W` reports swapping statistics, including `pswpin/s` and `pswpout/s`, which are the numbers of pages per second swapped in and out per second.

If `%memused` is near 100% and the scan rate is continuously over 200 pages per second, the system has a memory shortage.

Once a system runs out of real or physical memory and starts using swap space, its performance deteriorates dramatically. If you run out of swap space, your programs or the entire operating system are likely to crash. If `free` or `top` indicate that little swap space remains available, this is also an indication you are running low on memory.

The output from the `dmesg` command might include notification of any problems with physical memory that were detected at boot time.

9.2.2.3 Monitoring Block I/O Usage

The `iostat` command monitors the loading of block I/O devices by observing the time that the devices are active relative to the average data transfer rates. You can use this information to adjust the system configuration to balance the I/O loading across disks and host adapters.

`iostat -x` reports extended statistics about block I/O activity at one second intervals, including `%util`, which is the percentage of CPU time spent handling I/O requests to a device, and `avgqu-sz`, which is the average queue length of I/O requests that were issued to that device. If `%util` approaches 100% or `avgqu-sz` is greater than 1, device saturation is occurring.

You can also use the `sar -d` command to report on block I/O activity, including values for `%util` and `avgqu-sz`.

The `iotop` utility can help you identify which processes are responsible for excessive disk I/O. `iotop` has a similar user interface to `top`. In its upper section, `iotop` displays the total disk input and output usage in bytes per second. In its lower section, `iotop` displays I/O information for each process, including disk input output usage in bytes per second, the percentage of time spent swapping in pages from disk or waiting on I/O, and the command name. Use the left and right arrow keys to change the sort field, and press `A` to toggle the I/O units between bytes per second and total number of bytes, or `O` to toggle between displaying all processes or only those processes that are performing I/O.

9.2.2.4 Monitoring File System Usage

The `sar -v` command reports the number of unused cache entries in the directory cache (`dentunusd`) and the numbers of in-use file handles (`file-nr`), inode handlers (`inode-nr`), and pseudo terminals (`pty-nr`).

`iostat -n` reports I/O statistics for each NFS file system that is mounted.

9.2.2.5 Monitoring Network Usage

The `ip -s link` command displays network statistics and errors for all network devices, including the numbers of bytes transmitted (`TX`) and received (`RX`). The `dropped` and `overrun` fields provide an indicator of network interface saturation.

The `ss -s` command displays summary statistics for each protocol.

9.2.3 Using the Graphical System Monitor

The GNOME desktop environment includes a graphical system monitor that allows you to display information about the system configuration, running processes, resource usage, and file systems.

To display the System Monitor, use the following command:

```
# gnome-system-monitor
```

The **Resources** tab displays:

- CPU usage history in graphical form and the current CPU usage as a percentage.

- Memory and swap usage history in graphical form and the current memory and swap usage.

- Network usage history in graphical form, the current network usage for reception and transmission, and the total amount of data received and transmitted.

To display the System Monitor Manual, press `F1` or select **Help > Contents**.

9.2.4 About OSWatcher Black Box

Oracle OSWatcher Black Box (OSWbb) collects and archives operating system and network metrics that you can use to diagnose performance issues. OSWbb operates as a set of background processes on the server and gathers data on a regular basis, invoking such Unix utilities as `vmstat`, `mpstat`, `netstat`, `iostat`, and `top`.

OSWbb is particularly useful for Oracle RAC (Real Application Clusters) and Oracle Grid Infrastructure configurations. The RAC-DDT (Diagnostic Data Tool) script file includes OSWbb, but does not install it by default.

9.2.4.1 Installing OSWbb

To install OSWbb:

1. Log on to My Oracle Support (MOS) at http://support.oracle.com.

2. Download OSWatcher from the link listed by Doc ID 301137.1 at https://support.oracle.com/epmos/faces/DocumentDisplay?id=301137.1.

3. Copy the file to the directory where you want to install OSWbb, and run the following command:

```
# tar xvf oswbbVERS.tar
```

   `VERS` represents the version number of OSWatcher, for example 730 for OSWatcher 7.30.

   Extracting the tar file creates a directory named `oswbb`, which contains all the directories and files that are associated with OSWbb, including the `startOSWbb.sh` script.

4. To enable the collection of `iostat` information for NFS volumes, edit the `OSWatcher.sh` script in the `oswbb` directory, and set the value of `nfs_collect` to 1:

```
nfs_collect=1
```

9.2.4.2 Running OSWbb

To start OSWbb, run the `startOSWbb.sh` script from the `oswbb` directory.

```
# ./startOSWbb.sh [frequency duration]
```

The optional frequency and duration arguments specifying how often in seconds OSWbb should collect data and the number of hours for which OSWbb should run. The default values are 30 seconds and 48 hours. The following example starts OSWbb recording data at intervals of 60 seconds, and has it record data for 12 hours:

```
# ./startOSWbb.sh 60 12
...
Testing for discovery of OS Utilities...
VMSTAT found on your system.
IOSTAT found on your system.
MPSTAT found on your system.
IFCONFIG found on your system.
NETSTAT found on your system.
TOP found on your system.

Testing for discovery of OS CPU COUNT
oswbb is looking for the CPU COUNT on your system
CPU COUNT will be used by oswbba to automatically look for cpu problems

CPU COUNT found on your system.
CPU COUNT = 4

Discovery completed.

Starting OSWatcher Black Box v7.3.0  on date and time
With SnapshotInterval = 60
With ArchiveInterval = 12
...
Data is stored in directory: OSWbba_archive

Starting Data Collection...

oswbb heartbeat: date and time
oswbb heartbeat: date and time + 60 seconds
...
```

OSWbba_archive is the path of the archive directory that contains the OSWbb log files.

To stop OSWbb prematurely, run the `stopOSWbb.sh` script from the `oswbb` directory.

```
# ./stopOSWbb.sh
```

OSWbb collects data in the following directories under the `oswbb/archive` directory:

Directory	Description
oswifconfig	Contains output from `ifconfig`.
oswiostat	Contains output from `iostat`.
oswmeminfo	Contains a listing of the contents of `/proc/meminfo`.
oswmpstat	Contains output from `mpstat`.
oswnetstat	Contains output from `netstat`.
oswprvtnet	If you have enable private network tracing for RAC, contains information about the status of the private networks.
oswps	Contains output from `ps`.
oswslabinfo	Contains a listing of the contents of `/proc/slabinfo`.
oswtop	Contains output from `top`.
oswvmstat	Contains output from `vmstat`.

OSWbb stores data in hourly archive files named `system_name_utility_name_timestamp.dat`. Each entry in a file is preceded by a timestamp.

9.2.4.3 Analysing OSWbb Archived Files

From release v4.0.0, you can use the OSWbb analyzer (OSWbba) to provide information on system slowdowns, system hangs and other performance problems, and also to graph data collected from `iostat`, `netstat`, and `vmstat`. OSWbba requires that you have installed Java version 1.4.2 or higher on your system. You can use `yum` to install Java, or you can download a Java RPM for Linux from http://www.java.com.

Use the following command to run OSWbba from the `oswbb` directory:

```
# java -jar oswbba.jar -i OSWbba_archive
```

`OSWbba_archive` is the path of the archive directory that contains the OSWbb log files.

You can use OSWbba to display the following types of performance graph:

- Process run, wait and block queues.

- CPU time spent running in system, user, and idle mode.

- Context switches and interrupts.

- Free memory and available swap.

- Reads per second, writes per second, service time for I/O requests, and percentage utilization of bandwidth for a specified block device.

You can also use OSWbba to save the analysis to a report file, which reports instances of system slowdown,spikes in run queue length, or memory shortage, describes probable causes, and offers suggestions of how to improve performance.

```
# java -jar oswbba.jar -i OSWbba_archive -A
```

For more information about OSWbb and OSWbba, refer to the *OSWatcher Black Box User Guide* (Article ID 301137.1) and the *OSWatcher Black Box Analyzer User Guide* (Article ID 461053.1), which are available from My Oracle Support (MOS) at http://support.oracle.com.

Chapter 10 System Dump Analysis

Table of Contents

This chapter describes how to configure a system to create a memory image in the event of a system crash, and how to use the `crash` debugger to analyse the memory image in a crash dump or for a live system.

10.1 About Kdump

Kdump is the Linux kernel crash-dump mechanism. Oracle recommends that you enable the Kdump feature. In the event of a system crash, Kdump creates a memory image (`vmcore`) that can help in determining the cause of the crash. Enabling Kdump requires you to reserve a portion of system memory for exclusive use by Kdump. This memory is unavailable for other uses.

Kdump uses `kexec` to boot into a second kernel whenever the system crashes. `kexec` is a fast-boot mechanism which allows a Linux kernel to boot from inside the context of a kernel that is already running without passing through the bootloader stage.

10.1.1 Configuring and Using Kdump

During installation, you are given the option of enabling Kdump and specifying the amount of memory to reserve for it. If you prefer, you can enable kdump at a later time as described in this section.

If the `kexec-tools` and `system-config-kdump` packages are not already installed on your system, use `yum` to install them.

To enable Kdump by using the Kernel Dump Configuration GUI.

1. Enter the following command.

   ```
   # system-config-kdump
   ```

   The Kernel Dump Configuration GUI starts. If Kdump is currently disabled, the green **Enable** button is selectable and the **Disable** button is greyed out.

2. Click **Enable** to enable Kdump.

3. You can select the following settings tags to adjust the configuration of Kdump.

   Basic Settings   Allows you to specify the amount of memory to reserve for Kdump. The default setting is 128 MB.

Target Settings	Allows you to specify the target location for the `vmcore` dump file on a locally accessible file system, to a raw disk device, or to a remote directory using NFS or SSH over IPv4. The default location is `/var/crash`. You cannot save a dump file on an eCryptfs file system, on remote directories that are NFS mounted on the `rootfs` file system, or on remote directories that access require the use of IPv6, SMB, CIFS, FCoE, wireless NICs, multipathed storage, or iSCSI over software initiators to access them.
Filtering Settings	Allows to select which type of data to include in or exclude from the dump file. Selecting or deselecting the options alters the value of the argument that Kdump specifies to the `-d` option of the core collector program, `makedumpfile`.
Expert Settings	Allows you to choose which kernel to use, edit the command line options that are passed to the kernel and the core collector program, choose the default action if the dump fails, and modify the options to the core collector program, `makedumpfile`. The Unbreakable Enterprise Kernel supports the use of the `crashkernel=auto` setting for UEK Release 3 Quarterly Update 1 and later. If you use the `crashkernel=auto` setting, the output of the `dmesg` command shows `crashkernel=XM@0M`, which is normal. The setting actually reserves 128 MB plus 64 MB for each terabyte of physical memory.

> **Note**
>
> You cannot configure `crashkernel=auto` for Xen or for the UEK prior to UEK Release 3 Quarterly Update 1. Only standard settings such as `crashkernel=128M@48M` are supported. For systems with more than 128 GB of memory, the recommended setting is `crashkernel=512M@64M`.

You can select one of five default actions should the dump fail:

mount rootfs and run /sbin/init	Mount the root file system and run `init`. The `/etc/init.d/kdump` script attempts to save the dump to `/var/crash`, which requires a large amount of memory to be reserved.
reboot	Reboot the system, losing the `vmcore`. This is the default action.
shell	Enter a shell session inside the `initramfs` so that you can attempt to record the core. To reboot the system, exit the shell.
halt	Halt the system.

	poweroff		Power down the system.

Click **Help** for more information on these settings.

4. Click **Apply** to save your changes. The GUI displays a popup message to remind you that you must reboot the system for the changes to take effect.

5. Click **OK** to dismiss the popup messages.

6. Select **File > Quit**.

7. Reboot the system at a suitable time.

10.1.2 Files Used by Kdump

The Kernel Dump Configuration GUI modifies the following files:

File	Description
`/boot/grub2/grub.cfg`	Appends the `crashkernel` option to the kernel line to specify the amount of reserved memory and any offset value.
`/etc/kdump.conf`	Sets the location where the dump file can be written, the filtering level for the `makedumpfile` command, and the default behavior to take if the dump fails. See the comments in the file for information about the supported parameters.

If you edit these files, you must reboot the system for the changes to take effect.

For more information, see the `kdump.conf(5)` manual page.

10.1.3 Using Kdump with OCFS2

By default, a fenced node in an OCFS2 cluster restarts instead of panicking so that it can quickly rejoin the cluster. If the reason for the restart is not apparent, you can change the node's behavior so that it panics and generates a `vmcore` for analysis.

To configure a node to panic when it next fences, run the following command on the node after the cluster starts:

```
# echo panic > /sys/kernel/config/cluster/cluster_name/fence_method
```

where `cluster_name` is the name of the cluster. To set the value after each reboot of the system, add this line to `/etc/rc.local`. To restore the default behavior, set the value of `fence_method` to `reset` instead of `panic` and remove the line from `/etc/rc.local`.

For more information, see Section 23.3.5, "Configuring the Behavior of Fenced Nodes".

10.2 Using the crash Debugger

The `crash` utility allows you to analyze the state of the Oracle Linux system while it is running or of a core dump that resulted from a kernel crash. `crash` has been merged with the GNU Debugger `gdb` to provide source code debugging capabilities.

10.2.1 Installing the crash Packages

To use `crash`, you must install the `crash` package and the appropriate `debuginfo` and `debuginfo-common` packages.

To install the required packages:

1. Install the latest version of the `crash` package:

```
# yum install crash
```

2. Download the appropriate `debuginfo` and `debuginfo-common` packages for the `vmcore` or kernel that you want to examine from https://oss.oracle.com/ol6/debuginfo/:

   • If you want to examine the running Unbreakable Enterprise Kernel on the system, use commands such as the following to download the packages:

```
# export DLP="https://oss.oracle.com/ol6/debuginfo"
# wget ${DLP}/kernel-uek-debuginfo-`uname -r`.rpm
# wget ${DLP}/kernel-uek-debuginfo-common-`uname -r`.rpm
```

   • If you want to examine the running Red Hat Compatible Kernel on the system, use commands such as the following to download the packages:

```
# export DLP="https://oss.oracle.com/ol6/debuginfo"
# wget ${DLP}/kernel-debuginfo-`uname -r`.rpm
# wget ${DLP}/kernel-debuginfo-common-`uname -r`.rpm
```

   • If you want to examine a `vmcore` file that relates to a different kernel than is currently running, download the appropriate `debuginfo` and `debuginfo-common` packages for the kernel that produce the `vmcore`, for example:

```
# export DLP="https://oss.oracle.com/ol6/debuginfo"
# wget ${DLP}/kernel-uek-debuginfo-2.6.32-300.27.1.el6uek.x86_64.rpm
# wget ${DLP}/kernel-uek-debuginfo-common-2.6.32-300.27.1.el6uek.x86_64.rpm
```

> **Note**
>
> If the `vmcore` file was produced by Kdump, you can use the following `crash` command to determine the version:
>
> ```
> # crash --osrelease /var/tmp/vmcore/2013-0211-2358.45-host03.28.core
> 2.6.39-200.24.1.el6uek.x86_64
> ```

3. Install the `debuginfo` and `debuginfo-common` packages, for example:

```
# rpm -Uhv kernel-uek-debuginfo-2.6.32-300.27.1.el6uek.x86_64.rpm \
  kernel-uek-debuginfo-common-2.6.32-300.27.1.el6uek.x86_64.rpm
```

The `vmlinux` kernel object file (also known as the *namelist* file) that `crash` requires is installed in `/usr/lib/debug/lib/modules/kernel_version/`.

10.2.2 Running crash

> **Warning**
>
> Running `crash` on a live system is dangerous and can cause data corruption or total system failure. Do not use `crash` to examine a production system unless so directed by Oracle Support.

To examine the currently running kernel:

```
# crash
```

To determine the version of the kernel that produced a `vmcore` file:

```
# crash --osrelease /var/tmp/vmcore/2013-0211-2358.45-host03.28.core
2.6.39-200.24.1.el6uek.x86_64
```

To examine a `vmcore` file, specify the path to the file as an argument, for example:

```
# crash /var/tmp/vmcore/2013-0211-2358.45-host03.28.core
```

The appropriate `vmlinux` file must exist in `/usr/lib/debug/lib/modules/kernel_version/`.

If the `vmlinux` file is located elsewhere, specify its path before the path to the `vmcore` file, for example:

```
# crash /var/tmp/namelist/vmlinux-host03.28 /var/tmp/vmcore/2013-0211-2358.45-host03.28.core
```

The following `crash` output is from a `vmcore` file that was dumped after a system panic:

```
      KERNEL: /usr/lib/debug/lib/modules/2.6.39-200.24.1.el6uek.x86_64/vmlinux
    DUMPFILE: /var/tmp/vmcore/2013-0211-2358.45-host03.28.core
        CPUS: 2
        DATE: Fri Feb 11 16:55:41 2013
      UPTIME: 04:24;54
LOAD AVERAGE: 0.00, 0.01, 0.05
       TASKS: 84
    NODENAME: host03.mydom.com
     RELEASE: 2.6.39-200.24.1.el6uek.x86_64
     VERSION: #1 SMP Sat Jun 23 02:39:07 EDT 2012
     MACHINE: x86_64  (2992 MHz)
      MEMORY: 2 GB
       PANIC: "Oops: 0002" (check log for details)
         PID: 1696
     COMMAND: "insmod"
        TASK: c74de000
         CPU: 0
       STATE: TASK_RUNNING (PANIC)

crash>
```

The output includes the number of CPUs, the load average over the last 1 minute, last 5 minutes, and last 15 minutes, the number of tasks running, the amount of memory, the panic string, and the command that was executing at the time the dump was created. In this example, an attempt by `insmod` to install a module resulted in an *oops* violation.

At the crash> prompt, you can enter `help` or `?` to display the available `crash` commands. Enter `help command` to display more information for a specified command.

`crash` commands can be grouped into several different groups according to purpose:

Kernel Data Structure Analysis Commands	Display kernel text and data structures. See Section 10.2.3, "Kernel Data Structure Analysis Commands".
System state commands	Examine kernel subsystems on a system-wide or a per-task basis. See Section 10.2.4, "System State Commands".
Helper commands	Perform calculation, translation, and search functions. See Section 10.2.5, "Helper Commands"
Session control commands	Control the `crash` session. See Section 10.2.6, "Session Control Commands"

For more information, see the `crash(8)` manual page.

10.2.3 Kernel Data Structure Analysis Commands

The following `crash` commands takes advantage of `gdb` integration to display kernel data structures symbolically:

*   The *pointer-to* command can be used instead `struct` or `union`. The `gdb` module calls the appropriate function. For example:

```
crash> *buffer_head
struct buffer_head {
    long unsigned int b_state;
    struct buffer_head *b_this_page;
    struct page *b_page;
    sector_t b_blocknr;
    size_t b_size;
    char *b_data;
    struct block_device *b_bdev;
    bh_end_io_t *b_end_io;
    void *b_private;
    struct list_head b_assoc_buffers;
    struct address_space *b_assoc_map;
    atomic_t b_count;
}
SIZE: 104
```

dis   Disassembles source code instructions of a complete kernel function, from a specified address for a specified number of instructions, or from the beginning of a function up to a specified address. For example:

```
crash> dis fixup_irqs
0xffffffff81014486 <fixup_irqs>:        push    %rbp
0xffffffff81014487 <fixup_irqs+1>:      mov     %rsp,%rbp
0xffffffff8101448a <fixup_irqs+4>:      push    %r15
0xffffffff8101448c <fixup_irqs+6>:      push    %r14
0xffffffff8101448e <fixup_irqs+8>:      push    %r13
0xffffffff81014490 <fixup_irqs+10>:     push    %r12
0xffffffff81014492 <fixup_irqs+12>:     push    %rbx
0xffffffff81014493 <fixup_irqs+13>:     sub     $0x18,%rsp
0xffffffff81014497 <fixup_irqs+17>:     nopl    0x0(%rax,%rax,1)
...
```

p   Displays the contents of a kernel variable. For example:

```
crash> p init_mm
init_mm = $5 = {
  mmap = 0x0,
  mm_rb = {
    rb_node = 0x0
  },
  mmap_cache = 0x0,
  get_unmapped_area = 0,
  unmap_area = 0,
  mmap_base = 0,
  task_size = 0,
  cached_hole_size = 0,
  free_area_cache = 0,
  pgd = 0xffffffff81001000,
...
```

struct   Displays either a structure definition, or a formatted display of the contents of a structure at a specified address. For example:

```
crash> struct cpu
struct cpu {
    int node_id;
```

```
        int hotpluggable;
        struct sys_device sysdev;
}
SIZE: 88
```

sym   Translates a kernel symbol name to a kernel virtual address and section, or a kernel virtual
address to a symbol name and section. You can also query (`-q`) the symbol list for all symbols
containing a specified string or list (`-l`) all kernel symbols. For example:

```
crash> sym jiffies
ffffffff81b45880 (A) jiffies
crash> sym -q runstate
c590 (d) per_cpu__runstate
c5c0 (d) per_cpu__runstate_snapshot
ffffffff8100e563 (T) xen_setup_runstate_info
crash> sym -l
0 (D) __per_cpu_start
0 (D) per_cpu__irq_stack_union
4000 (D) per_cpu__gdt_page
5000 (d) per_cpu__exception_stacks
b000 (d) per_cpu__idt_desc
b010 (D) per_cpu__xen_cr0_value
b018 (D) per_cpu__xen_vcpu
b020 (D) per_cpu__xen_vcpu_info
b060 (d) per_cpu__mc_buffer
c570 (D) per_cpu__xen_mc_irq_flags
c578 (D) per_cpu__xen_cr3
c580 (D) per_cpu__xen_current_cr3
c590 (d) per_cpu__runstate
c5c0 (d) per_cpu__runstate_snapshot
...
```

union   Similar to the `struct` command, displaying kernel data types that are defined as unions instead
of structures.

whatis   Displays the definition of structures, unions, typedefs or text or data symbols. For example:

```
crash> whatis linux_binfmt
struct linux_binfmt {
    struct list_head lh;
    struct module *module;
    int (*load_binary)(struct linux_binprm *, struct pt_regs *);
    int (*load_shlib)(struct file *);
    int (*core_dump)(long int, struct pt_regs *, struct file *, long unsigned int);
    long unsigned int min_coredump;
    int hasvdso;
}
SIZE: 64
```

10.2.4 System State Commands

The following commands display kernel subsystems on a system-wide or per-task basis:

bt   Displays a kernel stack trace of the current context or of a specified PID or task. In the case of a
dump that followed a kernel panic, the command traces the functions that were called leading up
to the panic. For example:

```
crash> bt
PID: 10651  TASK: d1347000  CPU: 1   COMMAND: "insmod"
 #0 [d1547e44] die at c010785a
 #1 [d1547e54] do_invalid_op at c0107b2c
 #2 [d1547f0c] error_code (via invalid_op) at c01073dc
...
```

You can use the `-l` option to display the line number of the source file that corresponds to each function call in a stack trace.

```
crash> bt -l 1
PID: 1      TASK: ffff88007d032040  CPU: 1   COMMAND: "init"
 #0 [ffff88007d035878] schedule at ffffffff8144fdd4
    /usr/src/debug/kernel-2.6.32/linux-2.6.32.x86_64/kernel/sched.c: 3091
 #1 [ffff88007d035950] schedule_hrtimeout_range at ffffffff814508e4
    /usr/src/debug/kernel-2.6.32/linux-2.6.32.x86_64/arch/x86/include/asm/current.h: 14
 #2 [ffff88007d0359f0] poll_schedule_timeout at ffffffff811297d5
    /usr/src/debug/kernel-2.6.32/linux-2.6.32.x86_64/arch/x86/include/asm/current.h: 14
 #3 [ffff88007d035a10] do_select at ffffffff81129d72
    /usr/src/debug/kernel-2.6.32/linux-2.6.32.x86_64/fs/select.c: 500
 #4 [ffff88007d035d80] core_sys_select at ffffffff8112a04c
    /usr/src/debug/kernel-2.6.32/linux-2.6.32.x86_64/fs/select.c: 575
 #5 [ffff88007d035f10] sys_select at ffffffff8112a326
    /usr/src/debug/kernel-2.6.32/linux-2.6.32.x86_64/fs/select.c: 615
 #6 [ffff88007d035f80] system_call_fastpath at ffffffff81011cf2
    /usr/src/debug/////////kernel-2.6.32/linux-2.6.32.x86_64/arch/x86/kernel/entry_64.S:
    488
    RIP: 00007fce20a66243  RSP: 00007fff552c1038  RFLAGS: 00000246
    RAX: 0000000000000017  RBX: ffffffff81011cf2  RCX: ffffffffffffffff
    RDX: 00007fff552c10e0  RSI: 00007fff552c1160  RDI: 000000000000000a
    RBP: 0000000000000000   R8: 0000000000000000   R9: 0000000000000200
    R10: 00007fff552c1060  R11: 0000000000000246  R12: 00007fff552c1160
    R13: 00007fff552c10e0  R14: 00007fff552c1060  R15: 00007fff552c121f
    ORIG_RAX: 0000000000000017  CS: 0033  SS: 002b
```

`bt` is probably the most useful `crash` command. It has a large number of options that you can use to examine a kernel stack trace. For more information, enter `help bt`.

dev   Displays character and block device data. The `-d` and `-i` options display disk I/O statistics and I/O port usage. For example:

```
crash> dev
CHRDEV    NAME                CDEV         OPERATIONS
   1      mem            ffff88007d2a66c0  memory_fops
   4      /dev/vc/0      ffffffff821f6e30  console_fops
   4      tty            ffff88007a395008  tty_fops
   4      ttyS           ffff88007a3d3808  tty_fops
   5      /dev/tty       ffffffff821f48c0  tty_fops
...
BLKDEV    NAME                GENDISK      OPERATIONS
   1      ramdisk        ffff88007a3de800  brd_fops
 259      blkext             (none)
   7      loop           ffff880037809800  lo_fops
   8      sd             ffff8800378e9800  sd_fops
   9      md                 (none)
...
crash> dev -d
MAJOR GENDISK              NAME       REQUEST QUEUE       TOTAL ASYNC  SYNC   DRV
    8 0xffff8800378e9800 sda         0xffff880037b513e0     10    0     10     0
   11 0xffff880037cde400 sr0         0xffff880037b50b10      0    0      0     0
  253 0xffff880037902c00 dm-0        0xffff88003705b420      0    0      0     0
  253 0xffff880037d5f000 dm-1        0xffff88003705ab50      0    0      0     0
crash> dev -i
    RESOURCE       RANGE     NAME
ffffffff81a9e1e0  0000-ffff  PCI IO
ffffffff81a96e30  0000-001f  dma1
ffffffff81a96e68  0020-0021  pic1
ffffffff81a96ea0  0040-0043  timer0
ffffffff81a96ed8  0050-0053  timer1
ffffffff81a96f10  0060-0060  keyboard
...
```

files Displays information about files that are open in the current context or in the context of a specific PID or task. For example:

```
crash> files 12916
PID: 12916  TASK: ffff8800276a2480  CPU: 0   COMMAND: "firefox"
ROOT: /    CWD: /home/guest
  FD      FILE           DENTRY          INODE        TYPE PATH
   0 ffff88001c57ab00 ffff88007ac399c0 ffff8800378b1b68 CHR  /null
   1 ffff88007b315cc0 ffff88006046f800 ffff8800604464f0 REG  /home/guest/.xsession-errors
   2 ffff88007b315cc0 ffff88006046f800 ffff8800604464f0 REG  /home/guest/.xsession-errors
   3 ffff88001c571a40 ffff88001d605980 ffff88001be45cd0 REG  /home/guest/.mozilla/firefox
   4 ffff88003faa7300 ffff880063d83440 ffff88001c315bc8 SOCK
   5 ffff88003f8f6a40 ffff88007b41f080 ffff88007aef0a48 FIFO
...
```

fuser Displays the tasks that reference a specified file name or inode address as the current root directory, current working directory, open file descriptor, or that memory map the file. For example:

```
crash> fuser /home/guest
 PID         TASK        COMM            USAGE
 2990  ffff88007a2a8440  "gnome-session"  cwd
 3116  ffff8800372e6380  "gnome-session"  cwd
 3142  ffff88007c54e540  "metacity"       cwd
 3147  ffff88007aa1e440  "gnome-panel"    cwd
 3162  ffff88007a2d04c0  "nautilus"       cwd
 3185  ffff88007c00a140  "bluetooth-appl  cwd
...
```

irq Displays interrupt request queue data. For example:

```
crash> irq 0
    IRQ: 0
 STATUS: 400000 ()
HANDLER: ffffffff81b3da30          <ioapic_chip>
          typename: ffffffff815cdaef  "IO-APIC"
           startup: ffffffff8102a513  <startup_ioapic_irq>
          shutdown: ffffffff810aef92  <default_shutdown>
            enable: ffffffff810aefe3  <default_enable>
           disable: ffffffff810aeecc  <default_disable>
               ack: ffffffff8102a43d  <ack_apic_edge>
              mask: ffffffff81029be1  <mask_IO_APIC_irq>
...
```

kmem Displays the state of the kernel memory subsystems. For example:

```
crash> kmem -i
             PAGES        TOTAL       PERCENTAGE
 TOTAL MEM   512658       2 GB          ----
      FREE    20867     81.5 MB    4% of TOTAL MEM
      USED   491791      1.9 GB   95% of TOTAL MEM
    SHARED   176201    688.3 MB   34% of TOTAL MEM
   BUFFERS     8375     32.7 MB    1% of TOTAL MEM
    CACHED   229933    898.2 MB   44% of TOTAL MEM
      SLAB    39551    154.5 MB    7% of TOTAL MEM

TOTAL SWAP  1032190      3.9 GB         ----
 SWAP USED     2067      8.1 MB    0% of TOTAL SWAP
 SWAP FREE  1030123      3.9 GB   99% of TOTAL SWAP
```

kmem has a large number of options. For more information, enter help kmem.

log Displays the kernel message buffer in chronological order. This is the same data that dmesg displays but the output can include messages that never made it to syslog or disk.

---

mach   Displays machine-specific information such as the `cpuinfo` structure and the physical memory map.

mod   Displays information about the currently installed kernel modules. The `-s` and `-S` options load debug data (if available) from the specified module object files to enable symbolic debugging.

mount   Displays information about currently mounted file systems.

net   Displays network-related information.

ps   Displays information about processes. For example:

```
crash> ps Xorg crash bash
   PID    PPID  CPU        TASK       ST  %MEM      VSZ     RSS  COMM
   2679    2677  0  ffff88007cbcc400  IN   4.0   215488   84880  Xorg
> 13362  11853  0  ffff88007b25a500  RU   6.9   277632  145612  crash
   3685    3683  1  ffff880058714580  IN   0.1   108464    1984  bash
  11853   11845  1  ffff88001c6826c0  IN   0.1   108464    1896  bash
```

pte   Translates a page table entry (PTE) to the physical page address and page bit settings. If the PTE refers to a swap location, the command displays the swap device and offset.

runq   Displays the list of tasks that are on the run queue of each CPU.

sig   Displays signal-handling information for the current context or for a specified PID or task.

swap   Displays information about the configured swap devices.

task   Displays the contents of the `task_struct` for the current context or for a specified PID or task.

timer   Displays the entries in the timer queue in chronological order.

vm   Displays the virtual memory data, including the addresses of `mm_struct` and the page directory, resident set size, and total virtual memory size for the current context or for a specified PID or task.

vtop   Translates a user or kernel virtual address to a physical address. The command also displays the PTE translation, `vm_area_struct` data for user virtual addresses, `mem_map page` data for a physical page, and the swap location or file location if the page is not mapped.

waitq   Displays tasks that are blocked on a specified wait queue.

10.2.5 Helper Commands

The following commands perform calculation, translation, and search functions:

ascii   Translates a hexadecimal value to ASCII. With no argument, the command displays an ASCII chart.

btop   Translates a hexadecimal address to a page number.

eval   Evaluates an expression and displays the result in hexadecimal, decimal, octal, and binary. For example:

```
crash> eval 4g / 0x100
hexadecimal: 1000000   (16MB)
    decimal: 16777216
      octal: 100000000
     binary: 00000000000000000000000000000000000000000000000100000000000000000000000000000000
```

`list`   Displays the contents of a linked list of data objects, typically structures, starting at a specified address.

`ptob`   Translates a page number to its physical address (byte value).

`ptov`   Translates a physical address to a kernel virtual address.

`search`   Searches for a specified value in a specified range of user virtual memory, kernel virtual memory, or physical memory.

`rd`   Displays a selected range of user virtual memory, kernel virtual memory, or physical memory using the specified format.

`wr`   Writes a value to a memory location specified by symbol or address.

Warning

To avoid data loss or data corruption, take great care when using the `wr` command.

10.2.6 Session Control Commands

The following commands control the `crash` session:

`alias`   Defines an alias for a command. With no argument, the command displays the current list of aliases.

`exit`, `q`, or `quit`   Ends the `crash` session.

`extend`   Loads or unloads the specified `crash` extension shared object libraries.

`foreach`   Execute a `bt`, `files`, `net`, `task`, `set`, `sig`, `vm`, or `vtop` command on multiple tasks.

`gdb`   Passes any arguments to the GNU Debugger for processing.

`repeat`   Repeats a command indefinitely until you type `Ctrl-C`. This command is only useful when you use `crash` to examine a live system.

`set`   Sets the context to a specified PID or task. With no argument, the command displays the current context.

10.2.7 Guidelines for Examining a Dump File

The steps for debugging a memory dump from a kernel crash vary widely according to the problem. The following guidelines suggest some basic investigations that you can try:

- Use `bt` to trace the functions that led to the kernel panic.

- Use `bt -a` to trace the active task on each CPU. There is often a relationship between the panicking task on one CPU and the running tasks on the other CPUs. If the listed command is `cpu_idle` or `swapper`, no task was running on a CPU.

- Use `bt -l` to display the line number of the source files corresponding to each function call in the stack trace.

- Use `kmem -i` to obtain a summary of memory and swap usage. Look for a `SLAB` value greater than 500 MB and a `SWAP USED` value greater than 0%.

- Use `ps | grep UN` to check for processes in the `TASK_UNINTERRUPTIBLE` state (*D state*), usually because they are waiting on I/O. Such processes contribute to the load average and cannot be killed.

- Use `files` to display the files that a process had open.

You can shell indirection operators to save output from a command to a file for later analysis or to pipe the output through commands such as `grep`, for example:

```
crash> foreach files > files.txt
crash> foreach bt | grep bash
PID: 3685   TASK: ffff880058714580  CPU: 1   COMMAND: "bash"
PID: 11853  TASK: ffff88001c6826c0  CPU: 0   COMMAND: "bash"
```

Part II Networking and Network Services

This section contains the following chapters:

- Chapter 11, *Network Configuration* describes how to configure a system's network interfaces and network routing.

- Chapter 12, *Network Address Configuration* describes how to configure a DHCP server, DHCP client, and Network Address Translation.

- Chapter 13, *Name Service Configuration* describes how to use BIND to set up a DNS name server.

- Chapter 14, *Network Time Configuration* describes how to configure the chrony, Network Time Protocol (NTP), or Precision Time Protocol (PTP) daemons for setting the system time.

- Chapter 15, *Web Service Configuration* describes how to configure a basic HTTP server.

- Chapter 16, *Email Service Configuration* describes email programs and protocols that are available with Oracle Linux, and how to set up a basic Sendmail client.

- Chapter 17, *Load Balancing and High Availability Configuration* describes how to use Keepalived and HAProxy to set up load balancing and high availability configurations with networked systems.

- Chapter 18, *VNC Service Configuration* describes how to enable a VNC server to provide remote access to a graphical desktop.

Table of Contents

Chapter 11 Network Configuration

Table of Contents

This chapter describes how to configure a system's network interfaces and network routing.

11.1 About Network Interfaces

Each physical and virtual network device on an Oracle Linux system has an associated configuration file named `ifcfg-`*`interface`* in the `/etc/sysconfig/network-scripts` directory, where *`interface`* is the name of the interface. For example:

```
# cd /etc/sysconfig/network-scripts
# ls ifcfg-*
ifcfg-em1  ifcfg-em2  ifcfg-lo
```

In this example, there are two configuration files for motherboard-based Ethernet interfaces, `ifcfg-em1` and `ifcfg-em2`, and one for the loopback interface, `ifcfg-lo`. The system reads the configuration files at boot time to configure the network interfaces.

On your system, you might see other names for network interfaces. See Section 11.2, "About Network Interface Names".

The following are sample entries from an `ifcfg-em1` file for a network interface that obtains its IP address using the Dynamic Host Configuration Protocol (DHCP):

```
DEVICE="em1"
NM_CONTROLLED="yes"
ONBOOT=yes
USERCTL=no
TYPE=Ethernet
BOOTPROTO=dhcp
```

```
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="System em1"
UUID=5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03
HWADDR=08:00:27:16:C3:33
PEERDNS=yes
PEERROUTES=yes
```

If the interface is configured with a static IP address, the file contains entries such as the following:

```
DEVICE="em1"
NM_CONTROLLED="yes"
ONBOOT=yes
USERCTL=no
TYPE=Ethernet
BOOTPROTO=none
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="System em1"
UUID=5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03
HWADDR=08:00:27:16:C3:33
IPADDR=192.168.1.101
NETMASK=255.255.255.0
BROADCAST=192.168.1.255
PEERDNS=yes
PEERROUTES=yes
```

The following configuration parameters are typically used in interface configuration files:

BOOTPROTO	How the interface obtains its IP address:
	bootp Bootstrap Protocol (BOOTP).
	dhcp Dynamic Host Configuration Protocol (DHCP).
	none Statically configured IP address.
BROADCAST	IPv4 broadcast address.
DEFROUTE	Whether this interface is the default route.
DEVICE	Name of the physical network interface device (or a PPP logical device).
GATEWAYN	IPv4 gateway address for the interface. As an interface can be associated with several combinations of IP address, network mask prefix length, and gateway address, these are numbered starting from 0.
HWADDR	Media access control (MAC) address of an Ethernet device.
IPADDRN	IPv4 address of the interface.
IPV4_FAILURE_FATAL	Whether the device is disabled if IPv4 configuration fails.
IPV6_DEFAULTGW	IPv6 gateway address for the interface. For example: IPV6_DEFAULTGW=2001:0daa::2%em1.
IPV6_FAILURE_FATAL	Whether the device is disabled if IPv6 configuration fails.
IPV6ADDR	IPv6 address of the interface in CIDR notation, including the network mask prefix length. For example: IPV6ADDR="2001:0db8:1e11:115b::1/32"

`IPV6INIT`	Whether to enable IPv6 for the interface.
`MASTER`	Specifies the name of the master bonded interface, of which this interface is slave.
`NAME`	Name of the interface as displayed in the Network Connections GUI.
`NETWORK`	IPV4 address of the network.
`NM_CONTROLLED`	Whether the network interface device is controlled by the network management daemon, `NetworkManager`.
`ONBOOT`	Whether the interface is activated at boot time.
`PEERDNS`	Whether the `/etc/resolv.conf` file used for DNS resolution contains information obtained from the DHCP server.
`PEERROUTES`	Whether the information for the routing table entry that defines the default gateway for the interface is obtained from the DHCP server.
`PREFIX`*N*	Length of the IPv4 network mask prefix for the interface.
`SLAVE`	Specifies that this interface is a component of a bonded interface.
`TYPE`	Interface type.
`USERCTL`	Whether users other than `root` can control the state of this interface.
`UUID`	Universally unique identifier for the network interface device.

11.2 About Network Interface Names

Network interface names are based on information derived from the system BIOS or alternatively from a device's firmware, system path, or MAC address. This feature ensures that interface names persist across system reboots, hardware reconfiguration, and updates to device drivers and the kernel.

If you enable the `biosdevname` boot option (`biosdevname=1`), the `biosdevname` plugin to the udev device manager assigns names to network interfaces as follows:

- Ethernet interfaces on the motherboard are named `em`*N*, where *N* is the number of the interface starting from 1.

- Network interfaces on a PCI card are named `p`*S*`p`*P*, where *S* is the slot number and *P* is the port number.

- Virtual interfaces are named `p`*S*`p`*P*`_`*V*, where *S* is the slot number, *P* is the port number, and *V* is the virtual interface number.

If `biosdevname` is set to 0 (the default), `systemd` naming assigns the prefixes, `en`, `wl`, and `ww` to Ethernet, wireless LAN, and wireless WAN interfaces respectively. The prefix is followed by a suffix based on the hardware configuration, system bus configuration, or MAC address of the device:

`o`*N*	Onboard device with index number *N*.
`p`*B*`s`*S*`[f`*F*`][d`*D*`]`	PCI device with bus number *B*, slot number *S*, function number *F*, and device ID *D*.
`p`*B*`s`*S*`[f`*F*`][u`*P*`]...[c`*C*`][i`*I*`]`	USB device with bus number *B*, slot number *S*, function number *F*, port number *P*, configuration number *C*, and interface number *I*.

s*S*[f*F*][d*D*]   Hot-plug device with slot number *S*, function number *F*, and device ID *D*.

x*M*   Device with MAC address *M*.

For example, an Ethernet port on the motherboard might be named `eno1` or `em1`, depending on whether the value of `biosdevname` is 0 or 1.

The kernel assigns a legacy, unpredictable network interface name (`ethN` and `wlanN`) only if it cannot discover any information about the device that would allow it to disambiguate the device from other such devices. You can use the `net.ifnames=0` boot parameter to reinstate the legacy naming scheme.

> **Caution**
>
> Using the `net.ifnames` or `biosdevname` boot parameters to change the naming scheme can rendering existing firewall rules invalid. Changing the naming scheme can also affect other software that refers to network interface names.

11.3 About Network Configuration Files

The following sections describe additional network configuration files that you might need to configure on a system.

11.3.1 /etc/hosts

The `/etc/hosts` file associates host names with IP addresses. It allows the system to look up (`resolve`) the IP address of a host given its name, or the name given the IP address. Most networks use DNS (Domain Name Service) to perform address or name resolution. Even if your network uses DNS, it is usual to include lines in this file that specify the IPv4 and IPv6 addresses of the loopback device, for example:

```
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1          localhost localhost.localdomain localhost6 localhost6.localdomain6
```

The first and second column contains the IP address and host name. Additional columns contain aliases for the host name.

For more information, see the `hosts(5)` manual page.

11.3.2 /etc/nsswitch.conf

The `/etc/nsswitch.conf` file configures how the system uses various databases and name resolution mechanisms. The first field of entries in this file identifies the name of the database. The second field defines a list of resolution mechanisms in the order in which the system attempts to resolve queries on the database.

The following example hosts definition from `/etc/nsswitch.conf` indicates that the system first attempts to resolve host names and IP addresses by querying `files` (that is, `/etc/hosts`) and, if that fails, next by querying a DNS server, and last of all, by querying NIS+ (NIS version 3) :

```
hosts:      files dns nisplus
```

For more information, see the `nsswitch.conf(5)` manual page.

11.3.3 /etc/resolv.conf

The `/etc/resolv.conf` file defines how the system uses DNS to resolve host names and IP addresses. This file usually contains a line specifying the search domains and up to three lines that specify the IP

addresses of DNS server. The following entries from `/etc/resolv.conf` configure two search domains and three DNS servers:

```
search us.mydomain.com mydomain.com
nameserver 192.168.154.3
nameserver 192.168.154.4
nameserver 10.216.106.3
```

If your system obtains its IP address from a DHCP server, it is usual for the system to configure the contents of this file with information also obtained using DHCP.

For more information, see the `resolv.conf(5)` manual page.

11.3.4 /etc/sysconfig/network

The `/etc/sysconfig/network` file specifies additional information that is valid to all network interfaces on the system. The following entries from `/etc/sysconfig/network` define that IPv4 networking is enabled, IPv6 networking is not enabled, the host name of the system, and the IP address of the default network gateway:

```
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=host20.mydomain.com
GATEWAY=192.168.1.1
```

For more information, see `/usr/share/doc/initscripts*/sysconfig.txt`.

11.4 Command-line Network Configuration Interfaces

If the `NetworkManager` service is running, you can use the `nmcli` command to display the state of the system's physical network interfaces, for example:

```
# nmcli device status
DEVICE   TYPE      STATE
em1      ethernet  connected
em2      ethernet  connected
lo       loopback  unmanaged
```

You can use the `ip` command to display the status of an interface, for debugging, or for system tuning. For example, to display the status of all active interfaces:

```
# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:16:c3:33 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global em1
    inet6 fe80::a00:27ff:fe16:c333/64 scope link
       valid_lft forever preferred_lft forever
```

For each network interface, the output shows the current IP address, and the status of the interface. To display the status of a single interface such as `em1`, specify its name as shown here:

```
# ip addr show dev em1
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:16:c3:33 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global em1
    inet6 fe80::a00:27ff:fe16:c333/64 scope link
       valid_lft forever preferred_lft forever
```

You can also use `ip` to set properties and activate a network interface. The following example sets the IP address of the `em2` interface and activates it:

```
# ip addr add 10.1.1.1/24 dev em2
# ip link set em2 up
```

> **Note**
>
> You might be used to using the `ifconfig` command to perform these operations. However, `ifconfig` is considered obsolete and will eventually be replaced altogether by the `ip` command.

Any settings that you configure for network interfaces using `ip` do not persist across system reboots. To make the changes permanent, set the properties in the `/etc/sysconfig/network-scripts/ifcfg-interface` file.

Any changes that you make to an interface file in `/etc/sysconfig/network-scripts` do not take effect until you restart the network service or bring the interface down and back up again. For example, to restart the network service:

```
# systemctl restart network
```

To restart an individual interface, you can use the `ifup` or `ifdown` commands, which invoke the script in `/etc/sysconfig/network-scripts` that corresponds to the interface type, for example:

```
# ifdown em1
# ifup em1
Connection successfully activated
(D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/5)
```

Alternatively, you can use the `ip` command to stop and start network activity on an interface without completely tearing down and rebuilding its configuration:

```
# ip link set em1 down
# ip link set em1 up
```

The `ethtool` utility is useful for diagnosing potentially mismatched settings that affect performance, and allows you to query and set the low-level properties of a network device. Any changes that you make using `ethtool` do not persist across a reboot. To make the changes permanent, modify the settings in the device's `ifcfg-interface` file in `/etc/sysconfig/network-scripts`.

For more information, see the `ethtool(8)`, `ifup(8)`, `ip(8)`, and `nmcli(1)` manual pages.

11.5 Configuring Network Interfaces Using Graphical Interfaces

> **Note**
>
> The `NetworkManager` service and the `nmcli` command are included in the `NetworkManager` package. The Network Connections editor is included in the `nm-connection-editor` package.

The `NetworkManager` service dynamically detects and configures network connections. You can click on the network icon in the GNOME notification area to obtain information about the status of the network interfaces and to manage network connections:

• To enable or disable a network interface from the pull-down menu, use the On/Off toggle.

• To display the Settings window, select **Network Settings** from the drop-down menu.

Figure 11.2 shows the Network Settings editor.

Figure 11.1 Network Settings Editor

To create a new network interface, click **+**, select the interface type (**VPN**, **Bond**, **Bridge**, or **VLAN**). To edit an existing interface, select it from the list and click the gear icon.

Alternatively, you can use the Network Connections editor to configure wired, wireless, mobile broadband, Virtual Private Network (VPN), Digital Subscriber Link (DSL), and virtual (bond, bridge, team, and VLAN) interfaces. You can open this window by using the `nm-connection-editor` command. Figure 11.2 shows the Network Connections editor.

Figure 11.2 Network Connections Editor
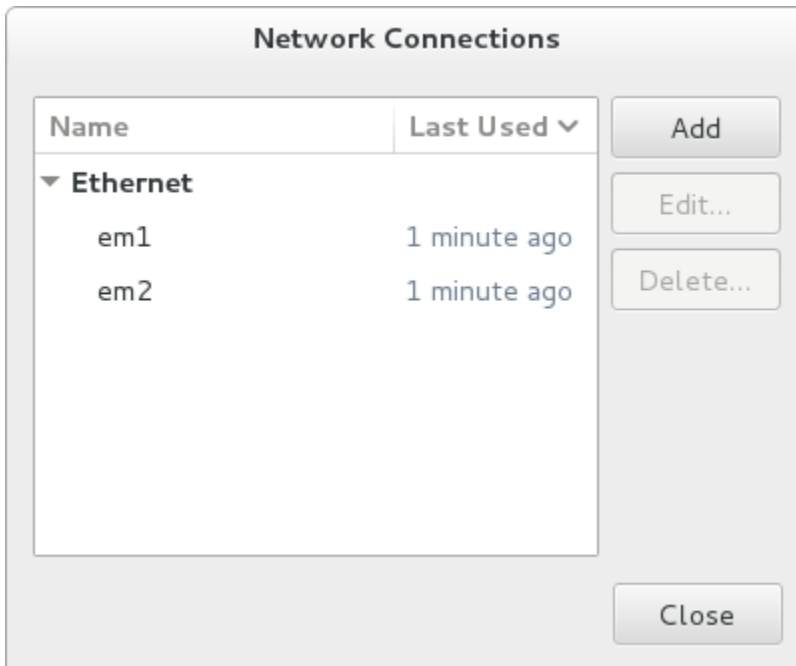
To create a new network interface, click **Add**, select the type of interface (hardware, virtual, or VPN) and click **Create**. To edit an existing interface, select it from the list and click **Edit**. To remove a selected interface, click **Delete**.

You can also use the `nmcli` command to manage network connections through `NetworkManager`. For more information, see the `nmcli(1)` manual page.

11.6 About Network Interface Bonding

Network interface bonding combines multiple network connections into a single logical interface. A bonded network interface can increase data throughput by load balancing or can provide redundancy by allowing failover from one component device to another. By default, a bonded interface appears like a normal network device to the kernel, but it sends out network packets over the available slave devices by using a simple round-robin scheduler. You can configure bonding module parameters in the bonded interface's configuration file to alter the behavior of load-balancing and device failover.

Basic load-balancing modes (`balance-rr` and `balance-xor`) work with any switch that supports EtherChannel or trunking. Advanced load-balancing modes (`balance-tlb` and `balance-alb`) do not impose requirements on the switching hardware, but do require that the device driver for each component interfaces implement certain specific features such as support for `ethtool` or the ability to modify the hardware address while the device is active. For more information see `/usr/share/doc/iputils-*/README.bonding`.

11.6.1 Configuring Network Interface Bonding

The bonding driver that is provided with the Oracle Linux kernel allows you to aggregate multiple network interfaces, such as `em1` and `em2`, into a single logical interface such as `bond0`. You can use the Network Settings editor to create the bond and then add network interfaces to this bond. Alternatively, you can use the `nmcli` command to create and configure the bond.

To create and configure a bonded interface from the command line:

1. Create the bond:

   ```
   # nmcli con add type bond con-name bond0 ifname bond0 mode balance-rr
   ```

   This example sets the name of the bond to `bond0` and its mode to `balance-rr`. For more information about the available options for load balancing or ARP link monitoring, see `/usr/share/doc/iputils-*/README.bonding` and the `nmcli(1)` manual page.

2. Add each interface to the bond:

   ```
   # nmcli con add type bond-slave ifname em1 master bond0
   # nmcli con add type bond-slave ifname em2 master bond0
   ```

   These commands add the `em1` and `em2` interfaces to `bond0`.

3. Restart the `NetworkManager` service:

   ```
   # systemctl restart NetworkManager
   ```

   After restarting the service, the bonded interface is available for use.

11.7 About Network Interface Teaming

> **Note**
>
> Network interface teaming requires Unbreakable Enterprise Kernel Release 3 (UEK R3) Quarterly Update 7 or later.

Network interface teaming is similar to network interface bonding and provides a way of implementing link aggregation that is relatively maintenance-free, as well as being simpler to modify, expand, and debug as compared with bonding.

A lightweight kernel driver implements teaming and the `teamd` daemon implements load-balancing and failover schemes termed *runners*. The following standard runners are defined:

`activebackup` Monitors the link for changes and selects the active port that is used to send packets.

`broadcast` Sends packets on all member ports.

`lacp` Provides load balancing by implementing the Link Aggregation Control Protocol 802.3ad on the member ports.

`loadbalance` In passive mode, uses the BPF hash function to select the port that is used to send packets.

In active mode, uses a balancing algorithm to distribute outgoing packets over the available ports.

`random` Selects a port at random to send each outgoing packet.

> **Note**
>
> UEK R3 does not currently support this runner mode.

`roundrobin` Transmits packets over the available ports in a round-robin fashion.

For specialized applications, you can create customized runners that `teamd` can interpret. The `teamdctl` command allows you to control the operation of `teamd`.

For more information, see the `teamd.conf(5)` manual page.

11.7.1 Configuring Network Interface Teaming

You can configure a teamed interface by creating JSON-format definitions that specify the properties of the team and each of its component interfaces. The `teamd` daemon then interprets these definitions. You can use the JSON-format definitions to create a team interface by starting the `teamd` daemon manually, by editing interface definition files in `/etc/sysconfig/network-scripts`, by using the `nmcli` command, or by using the Network Configuration editor (`nm-connection-editor`). This section describes the first of these methods.

To create a teamed interface by starting `teamd` manually:

1. Create a JSON-format definition file for the team and its component ports. For sample configurations, see the files under `/usr/share/doc/teamd-*/example_configs/`.

   The following example, which is based on the contents of the file `activebackup_ethtool_1.conf`, defines an active-backup configuration where `em4` is configured as the primary port and `em3` as the backup port and these ports are monitored by `ethtool`.

   ```
   {
           "device":        "team0",
           "runner":        {"name": "activebackup"},
           "link_watch":    {"name": "ethtool"},
           "ports":         {
                   "em3": {
   ```

```
                        "prio": -10,
                        "sticky": true
                },
                "em4": {
                        "prio": 100
                }
        }
}
```

2. Use the `ip` command to bring down the component ports:

```
# ip link set em3 down
# ip link set em4 down
```

   Active interfaces cannot be added to a team.

3. Start an instance of the `teamd` daemon and have it create the teamed interface by reading the configuration file (in this example, `/root/team_config/team0.conf`):

```
# teamd -g -f /root/team_config/team0.conf -d
Using team device "team0".
Using PID file "/var/run/teamd/team0.pid"
Using config file "/root/team_config/team0.conf"
```

   The `-g` option displays debugging messages and can be omitted.

4. Use the `ip` command to set the IP address and network mask prefix length of the teamed interface:

```
# ip addr add 10.0.0.5/24 dev team0
```

For more information, see the `teamd(8)` manual page.

11.7.2 Adding Ports to and Removing Ports from a Team

To add a port to a team, use the `teamdctl` command, for example:

```
# teamdctl team0 port add em5
```

To remove a port from a team:

```
# teamdctl team0 port remove em5
```

For more information, see the `teamdctl(8)` manual page.

11.7.3 Changing the Configuration of a Port in a Team

You can use the `teamdctl` command to update the configuration of a constituent port of a team, for example:

```
# teamdctl team0 port config update em3 '{"prio": -10, "sticky": false}'
```

Enclose the JSON-format definition in single quotes and do not split it over multiple lines.

For more information, see the `teamdctl(8)` manual page.

11.7.4 Removing a Team

To remove a team, use the following command to kill the `teamd` daemon:

```
# teamd -t team0 -k
```

For more information, see the `teamd(8)` manual page.

11.7.5 Displaying Information About Teams

To display the network state of the teamed interface, use the `ip` command:

```
# ip addr show dev team0
7: team0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 08:00:27:15:7a:f1 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.5/24 scope global team0
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe15:7af1/64 scope link
       valid_lft forever preferred_lft forever
```

You can use the `teamnl` command to display information about the component ports of the team:

```
# teamnl team0 ports
 5: em4: up 1000Mbit FD
 4: em3: up 1000Mbit FD
```

To display the current state of the team, use the `teamdctl` command, for example:

```
# teamdctl team0 state
setup:
  runner: activebackup
ports:
  em3
    link watches:
      link summary: down
      instance[link_watch_0]:
        name: ethtool
        link: down
  em4
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
runner:
  active port: em4
```

You can also use `teamdctl` to display the JSON configuration of the team and each of its constituent ports:

```
# teamdctl team0 config dump
{
    "device": "team0",
    "link_watch": {
        "name": "ethtool"
    },
    "mcast_rejoin": {
        "count": 1
    },
    "notify_peers": {
        "count": 1
    },
    "ports": {
        "em3": {
            "prio": -10,
            "sticky": true
        },
        "em4": {
            "prio": 100
        }
    },
```

```
    "runner": {
        "name": "activebackup"
    }
}
```

For more information, see the `teamdctl(8)` and `teamnl(8)` manual pages.

11.8 Configuring VLANs with Untagged Data Frames

A virtual local area network (VLAN) consists of a group of machines that can communicate as if they were attached to the same physical network. A VLAN allows you to group systems regardless of their actual physical location on a LAN. In a VLAN that uses untagged data frames, you create the broadcast domain by assigning the ports of network switches to the same permanent VLAN ID or PVID (other than 1, which is the default VLAN). All ports that you assign with this PVID are in a single broadcast domain. Broadcasts between devices in the same VLAN are not visible to other ports with a different VLAN, even if they exist on the same switch.

You can use the Network Settings editor or the `nmcli` command to create a VLAN device for an Ethernet interface.

To create a VLAN device from the command line, enter:

```
# nmcli con add type vlan con-name bond0-pvid10 ifname bond0-pvid10 dev bond0 id 10
```

This example sets up the VLAN device `bond0-pvid10` with a PVID of 10 for the bonded interface `bond0`. In addition to the regular interface, `bond0`, which uses the physical LAN, you now have a VLAN device, `bond0-pvid10`, which can use untagged frames to access the virtual LAN.

> **Note**
>
> You do not need to create virtual interfaces for the component interfaces of a bonded interface. However, you must set the PVID on each switch port to which they connect.

You can also use the command to set up a VLAN device for a non-bonded interface, for example:

```
# nmcli con add type vlan con-name em1-pvid5 ifname em1-pvid5 dev em1 id 5
```

To obtain information about the configured VLAN interfaces, view the files in the `/proc/net/vlan` directory.

11.8.1 Using the ip Command to Create VLAN Devices

The `ip` command provides an alternate method of creating VLAN devices. However, such devices do not persist across system reboots.

To create a VLAN interface `em1.5` for `em1` with a PVID of 5:

```
# ip link add link em1 name em1.5 type vlan id 5
```

For more information, see the `ip(8)` manual page.

11.9 Configuring Network Routing

A system uses its routing table to determine which network interface to use when sending packets to remote systems. If a system has only a single interface, it is sufficient to configure the IP address of a gateway system on the local network that routes packets to other networks.

To create a default route for IPv4 network packets, include an entry for GATEWAY in the `/etc/sysconfig/network` file. For example, the following entry configures the IP address of the gateway system:

```
GATEWAY=192.0.2.1
```

If your system has more than one network interface, you can specify which interface should be used:

```
GATEWAY=192.0.2.1
GATEWAYDEV=em1
```

A single statement is usually sufficient to define the gateway for IPv6 packets, for example:

```
IPV6_DEFAULTGW="2001:db8:1e10:115b::2%em1"
```

Any changes that you make to `/etc/sysconfig/network` do not take effect until you restart the network service:

```
# systemctl restart network
```

To display the routing table, use the `ip route show` command, for example:

```
# ip route show
10.0.2.0/24 dev em1  proto kernel  scope link  src 10.0.2.15
default via 10.0.2.2 dev em1  proto static
```

This example shows that packets destined for the local network (10.0.2.0/24) do not use the gateway. The default entry means that any packets destined for addresses outside the local network are routed via the gateway 10.0.2.2.

> **Note**
>
> You might be used to using the `route` command to configure routing. However, `route` is considered obsolete and will eventually be replaced altogether by the `ip` command.

You can also use the `netstat -rn` command to display this information:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.2.0        0.0.0.0         255.255.255.0   U         0 0          0 em1
0.0.0.0         10.0.2.2        0.0.0.0         UG        0 0          0 em1
```

To add or delete a route from the table, use the `ip route add` or `ip route del` commands. For example, to replace the entry for the static default route:

```
# ip route del default
# ip route show
10.0.2.0/24 dev em1  proto kernel  scope link  src 10.0.2.15
# ip ro add default via 10.0.2.1 dev em1 proto static
# ip route show
10.0.2.0/24 dev em1  proto kernel  scope link  src 10.0.2.15
default via 10.0.2.1 dev em1  proto static
```

To add a route to the network 10.0.3.0/24 via 10.0.3.1 over interface `em2`, and then delete that route:

```
# ip route add 10.0.4.0/24 via 10.0.2.1 dev em2
# ip route show
10.0.2.0/24 dev em1  proto kernel  scope link  src 10.0.2.15
10.0.3.0/24 via 10.0.3.1 dev em2
default via 10.0.2.2 dev em1  proto static
# ip route del 10.0.3.0/24
```

```
# ip route show
10.0.2.0/24 dev em1  proto kernel  scope link  src 10.0.2.15
default via 10.0.2.2 dev em1  proto static
```

The `ip route get` command is a useful feature that allows you to query the route on which the system will send packets to reach a specified IP address, for example:

```
# ip route get 23.6.118.140
23.6.118.140 via 10.0.2.2 dev em1  src 10.0.2.15
    cache  mtu 1500 advmss 1460 hoplimit 64
```

In this example, packets to 23.6.118.140 are sent out of the `em1` interface via the gateway 10.0.2.2.

Any changes that you make to the routing table using `ip route` do not persist across system reboots. To permanently configure static routes, you can configure them by creating a `route-interface` file in `/etc/sysconfig/network-scripts` for the interface. For example, you would configure a static route for the `em1` interface in a file named `route-em1`. An entry in these files can take the same format as the arguments to the `ip route add` command.

For example, to define a default gateway entry for `em1`, create an entry such as the following in `route-em1`:

```
default via 10.0.2.1 dev em1
```

The following entry in `route-em2` would define a route to 10.0.3.0/24 via 10.0.3.1 over `em2`:

```
10.0.3.0/24 via 10.0.3.1 dev em2
```

Any changes that you make to a `route-interface` file do not take effect until you restart either the network service or the interface.

For more information, see the `ip(8)` and `netstat(8)` manual pages.

Chapter 12 Network Address Configuration

Table of Contents

This chapter describes how to configure a DHCP server, DHCP client, and Network Address Translation.

12.1 About the Dynamic Host Configuration Protocol

The Dynamic Host Configuration Protocol (DHCP) allows client systems to obtain network configuration information from a DHCP server each time that they connect to the network. The DHCP server is configured with a range of IP addresses and other network configuration parameters that clients need.

When you configure an Oracle Linux system as a DHCP client, the client daemon, `dhclient`, contacts the DHCP server to obtain the networking parameters. As DHCP is broadcast-based, the client must be on the same subnet as either a server or a relay agent. If a client cannot be on the same subnet as the server, a DHCP relay agent can be used to pass DHCP messages between subnets.

The server provides a lease for the IP address that it assigns to a client. The client can request specific terms for the lease, such as the duration. You can configure a DHCP server to limit the terms that it can grant for a lease. Provided that a client remains connected to the network, `dhclient` automatically renews the lease before it expires. You can configure the DHCP server to provide the same IP address to a client based on the MAC address of its network interface.

The advantages of using DHCP include:

* centralized management of IP addresses

* ease of adding new clients to a network

* reuse of IP addresses reducing the total number of IP addresses that are required

* simple reconfiguration of the IP address space on the DHCP server without needing to reconfigure each client

For more information about DHCP, see RFC 2131.

12.2 Configuring a DHCP Server

To configure an Oracle Linux system as a DHCP server:

1. Install the `dhcp` package:

   ```
   #  yum install dhcp
   ```

2. Edit the `/etc/dhcp/dhcpd.conf` file to store the settings that the DHCP server can provide to the clients.

   The following example configures the domain name, a range of client addresses on the 192.168.2.0/24 subnet from 192.168.2.101 through 192.168.2.254 together with the IP addresses of the default

gateway and the DNS server, the default and maximum lease times in seconds, and a static IP address for the application server `svr01` that is identified by its MAC address:

```
option domain-name "mydom.org";
option domain-name-servers 192.168.2.1, 10.0.1.4;
option broadcast-address 192.168.2.255;
option routers 192.168.2.1;

subnet 192.168.2.0 netmask 255.255.255.0 {
  range 192.168.2.101 192.168.2.254;
  default-lease-time 10800;
  max-lease-time 43200;
}

host svr01 {
  hardware ethernet 80:56:3e:00:10:00;
  fixed-address 192.168.2.100;
  max-lease-time 86400;
}
```

The DHCP server sends the information in the `option` lines to each client when it requests a lease on an IP address. An option applies only to a subnet if you define it inside a `subnet` definition. In the example, the options are global and apply to both the `subnet` and `host` definitions. The `subnet` and `host` definitions have different settings for the maximum lease time.

> **Note**
>
> In Oracle Linux 7, the DHCP server no longer reads its configuration from `/etc/sysconfig/dhcpd`. Instead, it reads `/etc/dhcp/dhcpd.conf` to determine the interfaces on which it should listen.

For more information and examples, see `/usr/share/doc/dhcp-version/dhcpd.conf.sample` and the `dhcpd(8)` and `dhcp-options(5)` manual pages.

3. Touch the `/var/lib/dhcpd/dhcpd.leases` file, which stores information about client leases:

   ```
   # touch /var/lib/dhcpd/dhcpd.leases
   ```

4. Enter the following commands to start the DHCP service and ensure that it starts after a reboot:

   ```
   # systemctl start dhcpd
   # systemctl enable dhcpd
   ```

For information about configuring a DHCP relay, see the `dhcrelay(8)` manual page.

12.3 Configuring a DHCP Client

To configure an Oracle Linux system as a DHCP client:

1. Install the `dhclient` package:

   ```
   # yum install dhclient
   ```

2. Edit `/container/name/rootfs/etc/sysconfig/network-scripts/ifcfg-iface`, where `iface` is the name of the network interface, and change the value of `BOOTPROTO` to read as:

   ```
   BOOTPROTO=dhcp
   ```

3. Edit `/etc/sysconfig/network` and verify that it contains the following setting:

   ```
   NETWORKING=yes
   ```

4. To specify options for the client, such as the requested lease time and the network interface on which to request an address from the server, create the file `/etc/dhclient.conf` containing the required options.

   The following example specifies that the client should use the `em2` interface, request a lease time of 24 hours, and identify itself using its MAC address:

   ```
   interface "em2" {
     send dhcp-lease-time 86400;
     send dhcp-client-identifier 80:56:3e:00:10:00;
   }
   ```

   For more information, see the `dhclient.conf(5)` manual page.

5. Restart the network interface or the network service to enable the client, for example:

   ```
   # systemctl restart network
   ```

   When the client has requested and obtained a lease, information about this lease is stored in `/var/lib/dhclient/dhclient-interface.leases`.

For more information, see the `dhclient(8)` manual page.

12.4 About Network Address Translation

Network Address Translation (NAT) assigns a public address to a computer or a group of computers inside a private network with a different address scheme. The public IP address masquerades all requests as going to one server rather than several servers. NAT is useful for limiting the number of public IP addresses that an organization must finance, and for providing extra security by hiding the details of internal networks.

The `netfilter` kernel subsystem provides the `nat` table to implement NAT in addition to its tables for packet filtering. The kernel consults the `nat` table whenever it handles a packet that creates a new incoming or outgoing connection.

> **Note**
>
> If your want a system to be able to route packets between two of its network interfaces, you must turn on IP forwarding:
>
> ```
> # echo 1 > /proc/sys/net/ipv4/ip_forward
> ```

You can use the Firewall Configuration GUI (`firewall-config`) to configure masquerading and port forwarding.

Chapter 13 Name Service Configuration

Table of Contents

This chapter describes how to use BIND to set up a DNS name server.

13.1 About DNS and BIND

The Domain Name System (DNS) is a network-based service that maps (*resolves*) domain names to IP addresses. For a small, isolated network, you could use entries in the /etc/hosts file to provide the mapping, but most networks that are connected to the Internet use DNS.

DNS is a hierarchical and distributed database, where each level of the hierarchy is delimited by a period (`.`). Consider the following fully qualified domain name (FQDN):

```
wiki.us.mydom.com.
```

The root domain, represented by the final period in the FQDN, is usually omitted, except in DNS configuration files:

```
wiki.us.mydom.com
```

In this example, the top-level domain is `com`, `mydom` is a subdomain of `com`, `us` is a subdomain of `mydom`, and `wiki` is the host name. Each of these domains are grouped into zones for administrative purposes. A DNS server, or *name server*, stores the information that is needed to resolve the component domains inside a zone. In addition, a zone's DNS server stores pointers to the DNS servers that are responsible for resolving each subdomain.

If a client outside the `us.mydom.com` domain requests that its local name server resolve a FQDN such as `wiki.us.mydom.com` into an IP address for which the name server is not authoritative, the name server queries a root name server for the address of a name server that is authoritative for the `com` domain. Querying this name server returns the IP address of a name server for `mydom.com`. In turn, querying this name server returns the IP address of the name server for `us.oracle.com`, and querying this final name server returns the IP address for the FQDN. This process is known as a recursive query, where the local name server handles each referral from an external name server to another name server on behalf of the resolver.

Iterative queries rely on the resolver being able to handle the referral from each external name server to trace the name server that is authoritative for the FQDN. Most resolvers use recursive queries and so cannot use name servers that support only iterative queries. Fortunately, most

Oracle Linux provides the Berkeley Internet Name Domain (BIND) implementation of DNS. The `bind` package includes the DNS server daemon (`named`), tools for working with DNS such as `rndc`, and a number of configuration files, including:

`/etc/named.conf`	Contains settings for `named` and lists the location and characteristics of the zone files for your domain. Zone files are usually stored in `/var/named`.
`/etc/named.rfc1912.zones`	Contains several zone sections for resolving local loopback names and addresses.
`/var/named/named.ca`	Contains a list of the root authoritative DNS servers.

13.2 About Types of Name Servers

You can configure several types of name server using BIND, including:

Master name server	Authoritative for one or more domains, a master name server maintains its zone data in several database files, and can transfer this information periodically to any slave name servers that are also configured in the zone. In older documentation, master name servers are known as primary name servers. An organization might maintain two master name servers for a zone: one master outside the firewall to provide restricted information about the zone for publicly accessible hosts and services, and a hidden or *stealth* master inside the firewall that holds details of internal hosts and services.
Slave name server	Acting as a backup to a master name server, a slave name server maintains a copy of the zone data, which it periodically refreshes from the master's copy. In older documentation, slave name servers are known as secondary name servers.
Stub name server	A master name server for a zone might also be configured as a stub name server that maintains information about the master and slave name servers of child zones.
Caching-only name server	Performs queries on behalf of a client and stores the responses in a cache after returning the results to the client. It is not authoritative for any domains and the information that it records is limited to the results of queries that it has cached.
Forwarding name server	Forwards all queries to another name server and caches the results, which reduces local processing, external access, and network traffic.

In practice, a name server can be a combination of several of these types in complex configurations.

13.3 About DNS Configuration Files

Domains are grouped into zones and zones are configured through the use of zone files. Zone files store information about domains in the DNS database. Each zone file contains directives and resource records. Optional directives apply settings to a zone or instruct a name server to perform certain tasks. Resource records specify zone parameters and define information about the systems (*hosts*) in a zone.

For examples of BIND configuration files, see `/usr/share/doc/bind-version/sample/` .

13.3.1 /etc/named.conf

The main configuration file for `named` is `/etc/named.conf`, which contains settings for `named` and the top-level definitions for zones, for example:

128

```
include "/etc/rndc.key";

controls {
    inet 127.0.0.1 allow { localhost; } keys { "rndc-key"; }
};

zone "us.mydom.com" {
    type master;
    file "master-data";
    allow-update { key "rndc-key"; };
    notify yes;
};

zone "mydom.com" IN {
    type slave;
    file "sec/slave-data";
    allow-update { key "rndc-key"; };
    masters {10.1.32.1;};
};

zone "2.168.192.in-addr.arpa" IN {
    type master;
    file "reverse-192.168.2";
    allow-update { key "rndc-key"; };
    notify yes;
};
```

The `include` statement allows external files to be referenced so that potentially sensitive data such as key hashes can be placed in a separate file with restricted permissions.

The `controls` statement defines access information and the security requirements that are necessary to use the `rndc` command with the `named` server:

`inet`   Specifies which hosts can run `rndc` to control named. In this example, `rndc` must be run on the local host (127.0.0.1).

`keys`   Specifies the names of the keys that can be used. The example specifies using the key named `rndc-key`, which is defined in `/etc/rndc.key`. Keys authenticate various actions by `named` and are the primary method of controlling remote access and administration.

The `zone` statements define the role of the server in different zones.

The following zone options are used:

`type`   Specifies that this system is the master name server for the zone `us.mydom.com` and a slave server for `mydom.com`. `2.168.192.in-addr.arpa` is a reverse zone for resolving IP addresses to host names. See Section 13.3.3, "About Resource Records for Reverse-name Resolution ".

`file`   Specifies the path to the zone file relative to `/var/named`. The zone file for `us.mydom.com` is stored in `/var/named/master-data` and the transferred zone data for `mydom.com` is cached in `/var/named/sec/slave-data`.

`allow-update`   Specifies that a shared key must exist on both the master and a slave name server for a zone transfer to take place from the master to the slave. The following is an example record for a key in `/etc/rndc.key`:

```
key "rndc-key" {
    algorithm hmac-md5;
    secret "XQX8NmM41+RfbbSdcqOejg==";
};
```

You can use the `rndc-confgen -a` command to generate a key file.

`notify`	Specifies whether to notify the slave name servers when the zone information is updated.
`masters`	Specifies the master name server for a slave name server.

The next example is taken from the default `/etc/named.conf` file that is installed with the `bind` package, and which configures a caching-only name server.

```
options {
    listen-on port 53 { 127.0.0.1; };
    listen-on-v6 port 53 { ::1; };
    directory       "/var/named";
    dump-file       "/var/named/data/cache_dump.db";
    statistics-file "/var/named/data/named_stats.txt";
    memstatistics-file "/var/named/data/named_mem_stats.txt";
    allow-query { localnets; };
    recursion yes;

    dnssec-enable yes;
    dnssec-validation yes;
    dnssec-lookaside auto;

    /* Path to ISC DLV key */
    bindkeys-file "/etc/named.iscdlv.key";

    managed-keys-directory "/var/named/dynamic";
};

logging {
    channel default_debug {
        file "data/named.run";
        severity dynamic;
    };
};

zone "." IN {
    type hint;
    file "named.ca";
};

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

The `options` statement defines global server configuration options and sets defaults for other statements.

`listen-on`	The port on which `named` listens for queries.
`directory`	Specifies the default directory for zone files if a relative pathname is specified.
`dump-file`	Specifies where `named` dumps its cache if it crashes.
`statistics-file`	Specifies the output file for the `rndc stats` command.
`memstatistics-file`	Specifies the output file for `named` memory-usage statistics.
`allow-query`	Specifies which IP addresses may query the server. `localnets` specifies all locally attached networks.
`recursion`	Specifies whether the name server performs recursive queries.
`dnssec-enable`	Specifies whether to use secure DNS (DNSSEC).

`dnssec-validation`	Whether the name server should validate replies from DNSSEC-enabled zones.
`dnssec-lookaside`	Whether to enable DNSSEC Lookaside Validation (DLV) using the key in `/etc/named.iscdlv.key` defined by `bindkeys-file`.

The `logging` section enables logging of messages to `/var/named/data/named.run`. The `severity` parameter controls the logging level, and the `dynamic` value means that this level can be controlled by using the `rndc trace` command.

The `zone` section specifies the initial set of root servers using a hint zone. This zone specifies that `named` should consult `/var/named/named.ca` for the IP addresses of authoritative servers for the root domain (`.`).

For more information, see the `named.conf(5)` manual page and the BIND documentation in `/usr/share/doc/bind-version/arm`.

13.3.2 About Resource Records in Zone Files

A resource record in a zone file contains the following fields, some of which are optional depending on the record type:

Name	Domain name or IP address.
TTL (time to live)	The maximum time that a name server caches a record before it checks whether a newer one is available.
Class	Always `IN` for Internet.
Type	Type of record, for example:

`A` (address)	IPv4 address corresponding to a host.
`AAAA` (address)	IPv6 address corresponding to a host.
`CNAME` (canonical name)	Alias name corresponding to a host name.
`MX` (mail exchange)	Destination for email addressed to the domain.
`NS` (name server)	Fully qualified domain name of an authoritative name server for a domain.
`PTR` (pointer)	Host name corresponding to an IP address for address to name lookups (reverse-name resolution).
`SOA` (start of authority)	Authoritative information about a zone, such as the master name server, the email address of the domain's administrator, and the domain's serial number. All records following a `SOA` record relate to the zone that it defines up to the next `SOA` record.

Data	The information that the record stores, such as an IP address in an `A` record, or a host name in a `CNAME` or `PTR` record.

The following example shows the contents of a typical zone file such as `/var/named/master-data`:

```
$TTL 86400          ; 1 day
@ IN SOA dns.us.mydom.com. root.us.mydom.com. (
            57 ; serial
            28800 ; refresh (8 hours)
            7200 ; retry (2 hours)
            2419200 ; expire (4 weeks)
            86400 ; minimum (1 day)
            )
              IN   NS      dns.us.mydom.com.

dns            IN  A       192.168.2.1
us.mydom.com   IN  A       192.168.2.1
svr01          IN  A       192.168.2.2
www            IN  CNAME   svr01
host01         IN  A       192.168.2.101
host02         IN  A       192.168.2.102
host03         IN  A       192.168.2.103
...
```

A comment on a line is preceded by a semicolon (`;`).

The `$TTL` directive defines the default time-to-live value for all resource records in the zone. Each resource record can define its own time-to-live value, which overrides the global setting.

The `SOA` record is mandatory and included the following information:

`us.mydom.com`	The name of the domain.
`dns.us.mydom.com.`	The fully qualified domain name of the name server, including a trailing period (`.`) for the root domain.
`root.us.mydom.com.`	The email address of the domain administrator.
`serial`	A counter that, if incremented, tells `named` to reload the zone file.
`refresh`	The time after which a master name server notifies slave name servers that they should refresh their database.
`retry`	If a refresh fails, the time that a slave name server should wait before attempting another refresh.
`expire`	The maximum elapsed time that a slave name server has to complete a refresh before its zone records are no longer considered authoritative and it will stop answering queries.
`minimum`	The minimum time for which other servers should cache information obtained from this zone.

An `NS` record declares an authoritative name server for the domain.

Each `A` record specifies the IP address that corresponds to a host name in the domain.

The `CNAME` record creates the alias `www` for `svr01`.

For more information, see the BIND documentation in `/usr/share/doc/bind-version/arm`.

13.3.3 About Resource Records for Reverse-name Resolution

Forward resolution returns an IP address for a specified domain name. Reverse-name resolution returns a domain name for a specified IP address. DNS implements reverse-name resolution by using the special `in-addr.arpa` and `ip6.arpa` domains for IPv4 and IPv6.

The characteristics for a zone's `in-addr.arpa` or `ip6.arpa` domains are usually defined in `/etc/named.conf`, for example:

```
zone "2.168.192.in-addr.arpa" IN {
    type master;
    file "reverse-192.168.2";
    allow-update { key "rndc-key"; };
    notify yes;
};
```

The zone's name consists of `in-addr.arpa` preceded by the network portion of the IP address for the domain with its dotted quads written in reverse order.

If your network does not have a prefix length that is a multiple of 8, see RFC 2317 for the format that you should use instead.

The `PTR` records in `in-addr.arpa` or `ip6.arpa` domains define host names that correspond to the host portion of the IP address. The following example is take from the `/var/named/reverse-192.168.2` zone file:

```
$TTL 86400         ;
@ IN SOA dns.us.mydom.com. root.us.mydom.com. (
            57 ;
            28800 ;
            7200 ;
            2419200 ;
            86400 ;
            )
              IN   NS      dns.us.mydom.com.

1             IN   PTR     dns.us.mydom.com.
1             IN   PTR     us.mydom.com.
2             IN   PTR     svr01.us.mydom.com.
101           IN   PTR     host01.us.mydom.com.
102           IN   PTR     host02.us.mydom.com.
103           IN   PTR     host03.us.mydom.com.
...
```

For more information, see the BIND documentation in `/usr/share/doc/bind-version/arm`.

13.4 Configuring a Name Server

By default, the BIND installation allows you to configure a caching-only name server using the configuration settings that are provided in `/etc/named.conf` and files that it includes. This procedure assumes that you will either use the default settings or configure new `named` configuration and zone files.

To configure a name server:

1. Install the bind package:

    ```
    # yum install bind
    ```

2. If `NetworkManager` is enabled on the system, edit the `/etc/sysconfig/network-scripts/ifcfg-interface` file, and add the following entry:

    ```
    DNS1=127.0.0.1
    ```

    This line causes `NetworkManager` to add the following entry to `/etc/resolv.conf` when the network service starts:

    ```
    nameserver 127.0.0.1
    ```

This entry points the resolver at the local name server.

If you have disabled `NetworkManager`, edit `/etc/resolv.conf` to include the `nameserver 127.0.0.1` entry.

3. If required, modify the `named` configuration and zone files.

4. Configure the system firewall to allow incoming TCP connections to port 53 and incoming UDP datagrams on port 53:

```
# firewall-cmd --zone=zone --add-port=53/tcp --add-port=53/udp
# firewall-cmd --permanent --zone=zone --add-port=53/tcp --add-port=53/udp
```

5. Restart the `network` service, restart the `named` service, and configure `named` to start following system reboots:

```
# systemctl restart network
# systemctl start named
# systemctl enable named
```

13.5 Administering the Name Service

The `rndc` command allows you to administer the `named` service, either locally or from a remote machine (if permitted in the `controls` section of the `/etc/named.conf` file). To prevent unauthorized access to the service, `rndc` must be configured to listen on the selected port (by default, port 953), and both named and `rndc` must have access to the same key. To generate a suitable key, use the `rndc-confgen` command:

```
# rndc-confgen -a
wrote key file "/etc/rndc.key"
```

To ensure that only `root` can read the file:

```
# chmod o-rwx /etc/rndc.key
```

To check the status of the `named` service:

```
# rndc status
number of zones: 3
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
recursive clients: 0/1000
tcp clients: 0/100
server is up and running
```

If you modify the `named` configuration file or zone files, `rndc reload` instructs `named` to reload the files:

```
# rndc reload
server reload successful
```

For more information, see the `named(8)`, `rndc(8)` and `rndc-confgen(8)` manual pages.

13.6 Performing DNS Lookups

The `host` utility is recommended for performing DNS lookups. Without any arguments, `host` displays a summary of its command-line arguments and options. For example, look up the IP address for `host01`:

```
$ host host01
```

Perform a reverse lookup for the domain name that corresponds to an IP address:

```
$ host 192.168.2.101
```

Query DNS for the IP address that corresponds to a domain:

```
$ host dns.us.mydoc.com
```

Use the `-v` and `-t` options to display verbose information about records of a certain type:

```
$ host -v -t MX  www.mydom.com
Trying "www.mydom.com"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49643
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;www.mydom.com.    IN MX

;; ANSWER SECTION:
www.mydom.com.  135 IN CNAME www.mydom.com.acme.net.
www.mydom.com.acme.net. 1240 IN CNAME d4077.c.miscacme.net.

;; AUTHORITY SECTION:
c.miscacme.net. 2000 IN SOA m0e.miscacme.net. hostmaster.misc.com. ...

Received 163 bytes from 10.0.0.1#53 in 40 ms
```

The `-a` option (equivalent to `-v -t ANY`) displays all available records for a zone:

```
$ host -a www.us.mydom.com
Trying "www.us.mydom.com"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 40030
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.us.mydom.com.    IN ANY

;; ANSWER SECTION:
www.us.mydom.com.  263 IN CNAME www.us.mydom.acme.net.

Received 72 bytes from 10.0.0.1#53 in 32 ms
```

For more information, see the host(1) manual page.

Chapter 14 Network Time Configuration

Table of Contents

This chapter describes how to configure a system to use the chrony, Network Time Protocol (NTP), or Precision Time Protocol (PTP) daemons for setting the system time.

14.1 About the chronyd Daemon

The `chrony` package provides a `chronyd` service daemon and `chronyc` utility that enable mobile systems and virtual machines to update their system clock after a period of suspension or disconnection from a network.

The `chronyd` service is primarily designed to allow mobile systems and virtual machines to update their system clock after a period of suspension or disconnection from a network. However, you can also use it to implement a simple NTP client or a NTP server. When used as an NTP server, `chronyd` can synchronise with higher stratum NTP servers or it can act as a stratum 1 server using time signals received from the Global Positioning System (GPS) or radio broadcasts such as DCF77, MSF, or WWVB.

You can use the `chronyc` command to manage the `chronyd` service.

> **Note**
>
> `chronyd` uses NTP version 3 (RFC 1305), whose features are compatible with NTP version 4 (RFC 5905). However, `chronyd` does not support several important features of NTP version 4 nor does it support the use of PTP.

14.1.1 Configuring the chronyd Service

To configure the `chronyd` service on a system:

1. Install the `chrony` package.

   ```
   # yum install chrony
   ```

2. Edit `/etc/chrony.conf` to set up the configuration for `chronyd`.

   > **Note**
   >
   > The default configuration assumes that the system has network access to public NTP servers with which it can synchronise. The firewall rules for your internal networks might well prevent access to these servers but instead allow access to local NTP servers.

   The following example shows a sample configuration for a system that can access three NTP servers:

   ```
   server NTP_server_1
   ```

```
server NTP_server_2
server NTP_server_3
driftfile /var/lib/chrony/drift
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
```

The `commandkey` directive specifies the `keyfile` entry that `chronyd` uses to authenticate both `chronyc` commands and NTP packets. The `generatecommandkey` directive causes `chronyd` to generate an SHA1-based password automatically when the service starts.

To configure `chronyd` to act as an NTP server for a specified client or subnet, use the `allow` directive, for example:

```
server NTP_server_1
server NTP_server_2
server NTP_server_3
allow 192.168.2/24
driftfile /var/lib/chrony/drift
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
```

If a system has only intermittent access to NTP servers, the following configuration might be appropriate:

```
server NTP_server_1 offline
server NTP_server_2 offline
server NTP_server_3 offline
driftfile /var/lib/chrony/drift
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
```

If you specify the `offline` keyword, `chronyd` does not poll the NTP servers until it is told that network access is available. You can use the `chronyc -a online` and `chronyc -a offline` command to inform `chronyd` of the state of network access.

3. If remote access to the local NTP service is required, configure the system firewall to allow access to the NTP service in the appropriate zones, for example:

```
# firewall-cmd --zone=zone --add-service=ntp
success
# firewall-cmd --zone=zone --permanent --add-service=ntp
success
```

4. Start the `chronyd` service and configure it to start following a system reboot.

```
# systemctl start chronyd
# systemctl enable chronyd
```

You can use the `chronyc` command to display information about the operation of `chronyd` or to change its configuration, for example:

```
# chronyc -a
chrony version version
...
200 OK
chronyc> sources
210 Number of sources = 4
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^+ service1-eth3.debrecen.hp    2    6    37    21  -2117us[-2302us] +/-   50ms
```

```
^* ns2.telecom.lt              2   6    37    21   -811us[ -997us] +/-   40ms
^+ strato-ssd.vpn0.de          2   6    37    21   +408us[ +223us] +/-   78ms
^+ kvm1.websters-computers.c   2   6    37    22  +2139us[+1956us] +/-   54ms
chronyc> sourcestats
210 Number of sources = 4
Name/IP Address           NP  NR  Span  Frequency  Freq Skew  Offset   Std Dev
===============================================================================
service1-eth3.debrecen.hp  5   4   259    -0.394     41.803   -2706us    502us
ns2.telecom.lt             5   4   260    -3.948     61.422   +822us     813us
strato-ssd.vpn0.de         5   3   259     1.609     68.932   -581us     801us
kvm1.websters-computers.c  5   5   258    -0.263      9.586   +2008us    118us
chronyc> tracking
Reference ID    : 212.59.0.2 (ns2.telecom.lt)
Stratum         : 3
Ref time (UTC)  : Tue Sep 30 12:33:16 2014
System time     : 0.000354079 seconds slow of NTP time
Last offset     : -0.000186183 seconds
RMS offset      : 0.000186183 seconds
Frequency       : 28.734 ppm slow
Residual freq   : -0.489 ppm
Skew            : 11.013 ppm
Root delay      : 0.065965 seconds
Root dispersion : 0.007010 seconds
Update interval : 64.4 seconds
Leap status     : Normal
chronyc> exit
```

Using the `-a` option to `chronyc` is equivalent to entering the `authhash` and `password` subcommands, and avoids you having to specify the hash type and password every time that you use `chronyc`:

```
# cat /etc/chrony.keys

1 SHA1 HEX:4701E4D70E44B8D0736C8A862CFB6B8919FE340E
# chronyc
...
chronyc> authhash SHA1
chronyc> password HEX:4701E4D70E44B8D0736C8A862CFB6B8919FE340E
200 OK
```

For more information, see the `chrony(1)` and `chronyc(1)` manual pages, `/usr/share/doc/chrony-version/chrony.txt`, or use the `info chrony` command.

14.2 About the NTP Daemon

The `ntpd` daemon can synchronise the system clock with remote NTP servers, with local reference clocks, or with GPS and radio time signals. `ntpd` provides a complete implementation of NTP version 4 (RFC 5905) and is also compatibility with versions 3 (RFC 1305), 2 (RFC 1119), and 1 (RFC 1059).

You can configure `ntpd` to run in several different modes, as described at http://doc.ntp.org/4.2.6p5/assoc.html, using both symmetric-key and public-key cryptography, as described at http://doc.ntp.org/4.2.6p5/authopt.html.

14.2.1 Configuring the ntpd Service

To configure the `ntpd` service on a system:

1. Install the `ntp` package.

   ```
   # yum install ntp
   ```

2. Edit `/etc/ntp.conf` to set up the configuration for `ntpd`.

> **Note**
>
> The default configuration assumes that the system has network access to public NTP servers with which it can synchronise. The firewall rules for your internal networks might well prevent access to these servers but instead allow access to local NTP servers.

The following example shows a sample NTP configuration for a system that can access three NTP servers:

```
server NTP_server_1
server NTP_server_2
server NTP_server_3
server  127.127.1.0
fudge   127.127.1.0 stratum 10
driftfile /var/lib/ntp/drift
restrict default nomodify notrap nopeer noquery
```

The `server` and `fudge` entries for 127.127.1.0 cause `ntpd` to use the local system clock if the remote NTP servers are not available. The `restrict` entry allows remote systems only to synchronise their time with the local NTP service.

For more information about configuring `ntpd`, see http://doc.ntp.org/4.2.6p5/manyopt.html.

3. Create the drift file.

```
# touch /var/lib/ntp/drift
```

4. If remote access to the local NTP service is required, configure the system firewall to allow access to the NTP service in the appropriate zones, for example:

```
# firewall-cmd --zone=zone --add-service=ntp
success
# firewall-cmd --zone=zone --permanent --add-service=ntp
success
```

5. Start the `ntpd` service and configure it to start following a system reboot.

```
# systemctl start ntpd
# systemctl enable ntpd
```

You can use the `ntpq` and `ntpstat` commands to display information about the operation of `ntpd`, for example:

```
# ntpq -p
    remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*ns1.proserve.nl 193.67.79.202   2 u   21   64  377   31.420  10.742   3.689
-pomaz.hu        84.2.46.19      3 u   22   64  377   59.133  13.719   5.958
+server.104media 193.67.79.202   2 u   24   64  377   32.110  13.436   5.222
+public-timehost 193.11.166.20   2 u   28   64  377   57.214   9.304   6.311
# ntpstat
synchronised to NTP server (80.84.224.85) at stratum 3
   time correct to within 76 ms
   polling server every 64
```

For more information, see the `ntpd(8)`, `ntpd.conf(5)`, `ntpq(8)`, and `ntpstat(8)` manual pages and http://doc.ntp.org/4.2.6p5/.

14.3 About PTP

PTP allows you to synchronise system clocks on a local area network to a higher accuracy than NTP. Provided that network drivers support either hardware or software time stamping, a PTP clock can use the time stamps in PTP messages to compensate for propagation delays across a network. Software time stamping allows PTP to synchronise systems to within a few tens of microseconds. With hardware time stamping, PTP can synchronise systems to within a few tenths of a microsecond. If you require high-precision time synchronization of systems, use hardware time stamping. Both the UEK R3 and RHCK kernels support PTP version 2 as defined in IEEE 1588.

A typical PTP configuration on an enterprise local area network consists of:

- One or more *grandmaster clock* systems.

  A grandmaster clock is typically implemented as specialized hardware that can use high-accuracy GPS signals or lower-accuracy code division multiple access (CDMA) signals, radio clock signals, or NTP as a time reference source. If several grandmaster clocks are available, the best master clock (BMC) algorithm selects the grandmaster clock based on the settings of their `priority1`, `clockClass`, `clockAccuracy`, `offsetScaledLogVariance`, and `priority2` parameters and their unique identifier, in that order.

- Several *boundary clock* systems.

  Each boundary clock is slaved to a grandmaster clock on one subnetwork and relays PTP messages to one or more additional subnetworks. A boundary clock is usually implemented as a function of a network switch.

- Multiple *slave clock* systems.

  Each slave clock on a subnetwork is slaved to a boundary clock, which acts as the *master clock* for that slave clock.

A simpler configuration is to set up a single grandmaster clock and multiple slave clocks on the same network segment, which removes any need for an intermediate layer of boundary clocks.

Grandmaster and slave clock systems that use only one network interface for PTP are termed *ordinary clocks*.

Boundary clocks require at least two network interfaces for PTP: one interface acts a slave to a grandmaster clock or a higher-level boundary clock; the other interfaces act as masters to slave clocks or lower-level boundary clocks.

Synchronization of boundary and slave clock systems is achieved by sending time stamps in PTP messages. By default, PTP messages are sent in UDPv4 datagrams. It is also possible to configure PTP to use UDPv6 datagrams or Ethernet frames as its transport mechanism.

To be able to use PTP with a system, the driver for at least one of the system's network interfaces must support either software or hardware time stamping. To find out whether the driver for a network interface supports time stamping, use the `ethtool` command as shown in the following example:

```
# ethtool -T em1
Time stamping parameters for em1:
Capabilities:
 hardware-transmit     (SOF_TIMESTAMPING_TX_HARDWARE)
 software-transmit     (SOF_TIMESTAMPING_TX_SOFTWARE)
 hardware-receive      (SOF_TIMESTAMPING_RX_HARDWARE)
 software-receive      (SOF_TIMESTAMPING_RX_SOFTWARE)
 software-system-clock (SOF_TIMESTAMPING_SOFTWARE)
```

```
hardware-raw-clock     (SOF_TIMESTAMPING_RAW_HARDWARE)
...
```

The output from `ethtool` in this example shows that the `em1` interface supports both hardware and software time stamping capabilities.

With software time stamping, `ptp4l` synchronises the system clock to an external grandmaster clock.

If hardware time stamping is available, `ptp4l` can synchronise the PTP hardware clock to an external grandmaster clock. In this case, you use the `phc2sys` daemon to synchronise the system clock with the PTP hardware clock.

14.3.1 Configuring the PTP Service

To configure the PTP service on a system:

1. Install the `linuxptp` package.

    ```
    # yum install linuxptp
    ```

2. Edit `/etc/sysconfig/ptp4l` and define the start-up options for the `ptp4l` daemon.

    Grandmaster clocks and slave clocks require that you define only one interface.

    For example, to use hardware time stamping with interface `em1` on a slave clock:

    ```
    OPTIONS="-f /etc/ptp4l.conf -i em1 -s"
    ```

    To use software time stamping instead of hardware time stamping, specify the `-S` option:

    ```
    OPTIONS="-f /etc/ptp4l.conf -i em1 -S -s"
    ```

    > **Note**
    >
    > The `-s` option specifies that the clock operates only as a slave (`slaveOnly` mode). Do not specify this option for a grandmaster clock or a boundary clock.

    For a grandmaster clock, omit the `-s` option, for example:

    ```
    OPTIONS="-f /etc/ptp4l.conf -i em1"
    ```

    A boundary clock requires that you define at least two interfaces, for example:

    ```
    OPTIONS="-f /etc/ptp4l.conf -i em1 -i em2"
    ```

    You might need to edit the file `/etc/ptp4l.conf` to make further adjustments to the configuration of `ptp4l`, for example:

    *   For a grandmaster clock, set the value of the `priority1` parameter to a value between 0 and 127, where lower values have higher priority when the BMC algorithm selects the grandmaster clock. For a configuration that has a single grandmaster clock, a value of 127 is suggested.

    *   If you set the value of `summary_interval` to an integer value N instead of 0, `ptp4l` writes summary clock statistics to `/var/log/messages` every 2^N seconds instead of every second ($2^0 = 1$). For example, a value of 10 would correspond to an interval of 2^{10} or 1024 seconds.

    *   The `logging_level` parameter controls the amount of logging information that `ptp4l` records. The default value of `logging_level` is 6, which corresponds to `LOG_INFO`. To turn off logging completely, set the value of `logging_level` to 0. Alternatively, specify the `-q` option to `ptp4l`.

For more information, see the `ptp4l(8)` manual page.

3. Configure the system firewall to allow access by PTP event and general messages to UDP ports 319 and 320 in the appropriate zone, for example:

```
# firewall-cmd --zone=zone --add-port=319/udp --add-port=320/udp
success
# firewall-cmd --permanent --zone=zone --add-port=319/udp --add-port=320/udp
success
```

4. Start the `ptp4l` service and configure it to start following a system reboot.

```
# systemctl start ptp4l
# systemctl enable ptp4l
```

5. To configure `phc2sys` on a clock system that uses hardware time stamping:

   a. Edit `/etc/sysconfig/phc2sys` and define the start-up options for the `phc2sys` daemon.

   On a boundary clock or slave clock, synchronise the system clock with the PTP hardware clock that is associated with the slave network interface, for example:

   ```
   OPTIONS="-c CLOCK_REALTIME -s em1 -w"
   ```

   Note

   The slave network interface on a boundary clock is the one that it uses to communicate with the grandmaster clock.

   The `-w` option specifies that `phc2sys` waits until `ptp4l` has synchronised the PTP hardware clock before attempting to synchronise the system clock.

   On a grandmaster clock, which derives its system time from a reference time source such as GPS, CDMA, NTP, or a radio time signal, synchronise the network interface's PTP hardware clock from the system clock, for example:

   ```
   OPTIONS="-c em1 -s CLOCK_REALTIME -w"
   ```

   For more information, see the `phc2sys(8)` manual page.

   b. Start the `phc2sys` service and configure it to start following a system reboot.

   ```
   # systemctl start phc2sys
   # systemctl enable phc2sys
   ```

You can use the `pmc` command to query the status of `ptp4l` operation. The following example shows the results of running `pmc` on a slave clock system that is directly connected to the grandmaster clock system without any intermediate boundary clocks:

```
# pmc -u -b 0 'GET TIME_STATUS_NP'
sending: GET TIME_STATUS_NP
 080027.fffe.7f327b-0 seq 0 RESPONSE MANAGEMENT TIME_STATUS_NP
  master_offset              -98434
  ingress_time               1412169090025854874
  cumulativeScaledRateOffset +1.000000000
  scaledLastGmPhaseChange    0
  gmTimeBaseIndicator        0
  lastGmPhaseChange          0x0000'0000000000000000.0000
  gmPresent                  true
  gmIdentity                 080027.fffe.d9e453
# pmc -u -b 0 'GET CURRENT_DATA_SET'
```

```
sending: GET CURRENT_DATA_SET
 080027.fffe.7f327b-0 seq 0 RESPONSE MANAGEMENT CURRENT_DATA_SET
  stepsRemoved      1
  offsetFromMaster  42787.0
  meanPathDelay     289207.0
```

Useful information in this output includes:

gmIdentity
: The unique identifier of the grandmaster clock, which is based on the MAC address of its network interface.

gmPresent
: Whether an external grandmaster clock is available. This value is displayed as `false` on the grandmaster clock itself.

meanPathDelay
: An estimate of how many nanoseconds by which synchronization messages are delayed.

offsetFromMaster
: The most recent measurement of the time difference in nanoseconds relative to the grandmaster clock.

stepsRemoved
: The number of network steps between this system and the grandmaster clock.

For more information, see the `phc2sys(8)`, `pmc(8)`, and `ptp4l(8)` manual pages, http://www.zhaw.ch/en/engineering/institutes-centres/ines/downloads/documents.html, and IEEE 1588.

14.3.2 Using PTP as a Time Source for NTP

To make the PTP-adjusted system time on an NTP server available to NTP clients, include the following entries in `/etc/ntp.conf` on the NTP server to define the local system clock as the time reference:

```
server     127.127.1.0
fudge      127.127.1.0 stratum 0
```

Note

Do not configure any additional `server` lines in the file.

For more information, see Section 14.2.1, "Configuring the ntpd Service".

Chapter 15 Web Service Configuration

Table of Contents

This chapter describes how to configure a basic HTTP server.

15.1 About the Apache HTTP Server

Oracle Linux provides the Apache HTTP Server, which is an open-source web server developed by the Apache Software Foundation. The Apache server hosts web content, and responds to requests for this content from web browsers such as Firefox.

15.2 Installing the Apache HTTP Server

To install the Apache HTTP server:

1. Enter the following command:

```
# yum install httpd
```

2. Start the server, and configure it to start after system reboots:

```
# apachectl start
# systemctl enable httpd
```

3. Check for configuration errors:

```
# apachectl configtest
```

4. Create firewall rules to allow access to the ports on which the HTTP server listens, for example:

```
# firewall-cmd --zone=zone --add-service=http
# firewall-cmd --permanent --zone=zone --add-service=http
```

15.3 Configuring the Apache HTTP Server

> **Note**
>
> Any changes that you make to the configuration of the Apache HTTP server do not take effect until you restart the server:
>
> ```
> # apachectl restart
> ```

The main configuration file for the Apache HTTP server is `/etc/httpd/conf/httpd.conf`. You can modify the directives in this file to customize Apache for your environment.

The directives include:

`Allow from client [client ...]	all`	Specifies a list of clients that can access content or `all` to serve content to any client. The `Order` directive determines the order in which `httpd` evaluates `Allow` and `Deny` directives.
`Deny from client [client ...]	all`	Specifies a list of clients that cannot access content or `all` to disallow all clients. The `Order` directive determines the order in which `httpd` evaluates `Allow` and `Deny` directives.

`DocumentRoot directory-path`

The top level directory for Apache server content. The `apache` user requires read access to any files and read and execute access to the directory and any of its sub-directories. Do not place a slash at the end of the directory path.

For example:

```
DocumentRoot /var/www/html
```

If you specify a different document root or link to content that is not under `/var/www/html` and SELinux is enabled in enforcing mode on your system, change the default file type of the directory hierarchy that contains the content to `httpd_sys_content_t`:

1. Use the `semanage` command to define the default file type of the content directory as `httpd_sys_content_t`:

   ```
   # /usr/sbin/semanage fcontext -a -t httpd_sys_content_t "content_dir(/.*)?"
   ```

2. Use the `restorecon` command to apply the file type to the entire content directory hierarchy.

   ```
   # /sbin/restorecon -R -v content_dir
   ```

`ErrorLog filename | syslog[:facility]`

If set to a file name, specifies the file, relative to `ServerRoot`, to which `httpd` sends error messages.

If set to `syslog`, specifies that `httpd` send errors to `rsyslogd`. A `facility` argument specifies the `rsyslogd` facility. The default facility is `local7`.

For example:

```
ErrorLog logs/error_log
```

`Listen [IP_address:]port`

Accept incoming requests on the specified port or IP address and port combination. By default, the `httpd` server accepts requests on port 80 for all network interfaces. For a port number other than 80, HTTP requests to the server must include the port number.

For example:

```
Listen 80
Listen 192.168.2.1:8080
```

`LoadModule module path`

The Apache HTTP server can load external modules (dynamic shared objects or DSOs) to extend its functionality. The `module` argument is the name of the DSO, and `filename` is the path name of the module relative to `ServerRoot`.

For example:

```
LoadModule auth_basic_module modules/mod_auth_basic.so
```

Order deny,allow | allow,deny

Specifies the order in which `httpd` evaluates Allow and Deny directives.

For example, permit access only to clients from the `mydom.com` domain:

```
Order deny,allow
Deny from all
Allow from .mydom.com
```

The following directives would not permit access by any client:

```
Order allow,deny
Deny from all
Allow from .mydom.com
```

ServerName *FQDN*[:*port*]

Specifies the fully qualified domain name or IP address of the `httpd` server and an optional port on which the server listens. The FQDN must be resolvable to an IP address. If you do not specify a FQDN, the server performs a reverse-name lookup on the IP address. If you do not specify a port, the server uses the port corresponding to the incoming request.

For example:

```
ServerName www.mydom.com:80
```

ServerRoot *directory-path*

The top of the directory hierarchy where the `httpd` server keeps its configuration, error, and log files. Do not place a slash at the end of the directory path.

For example:

```
ServerRoot /etc/httpd
```

Timeout *seconds*

Specifies the number of seconds that `httpd` waits for network operations to finish before reporting a timeout error. The default value is 60 seconds.

UserDir *directory-path* ... | disabled [*user* ...] | enabled *user* ...

If set to `disabled`, disallows users identified by the space-separated *user* argument to publish content from their home directories. If no users are specified, all users are disallowed.

If set to `enabled`, allows users identified by the space-separated *user* argument to publish content from their home directories, provided that they are not specified as an argument to `disabled`.

directory-path is the name of a directory from which `httpd` publishes content. A relative path is assumed to be relative to a user's home directory. If you specify more than one directory path, `httpd` tries each alternative in turn until find a web page. If *directory-path* is not defined, the default is `~/public_html`. Do not place a slash at the end of the directory path.

For example:

```
UserDir disabled root guest
UserDir enabled oracle alice
UserDir www http://www.mydom.com/
```

The `root` and `guest` users are disabled from content publishing. Assuming that `ServerName` is set to `www.mydom.com`, browsing `http://www.example.com/~alice` displays `alice`'s web page, which must be located at `~alice/www` or `http://www.example.com/alice` (that is, in the directory `alice` relative to `ServerRoot`).

> **Note**
>
> You would usually change the settings in the `<IfModule mod_userdir.c>` container to allow users to publish user content.

For more information, see http://httpd.apache.org/docs/current/mod/directives.html.

15.4 Testing the Apache HTTP Server

To test that an Apache HTTP server is working:

- From the local system, direct a browser on the local system to `http://localhost`.

- From a remote system, direct a browser to `http://` followed by the value of the `ServerName` directive specified in the configuration file (`/etc/httpd/conf/httpd.conf`).

If the browser displays the Apache 2 Test Page, the server is working correctly.

To test that the server can deliver content, create an HTML file named `index.html` in the directory specified by the `DocumentRoot` directive (by default, `/var/www/html`). After reloading the page, the browser should display this HTML file instead of the Apache 2 Test Page.

15.5 Configuring Apache Containers

Apache containers are special directives that group other directives, often to create separate web directory hierarchies with different characteristics. A container is delimited by the XML-style tags `<type>` and `</type>`, where `type` is the container type.

The following are examples of container types:

`<Directory directory-path>`

Applies the contained directives to directories under `directory-path`. The following example applies the `Deny`, `Allow`, and `AllowOverride` directives to all files and directories under `/var/www/html/sandbox`.

```
<Directory /var/www/html/sandbox>
  Deny from all
  Allow from 192.168.2.
  AllowOverride All
</Directory>
```

The `AllowOverride` directive is only used in `Directory` containers and specifies which classes of directives are allowed in `.htaccess` files. (`.htaccess` configuration files typically contain user

authentication directives for a web directory.) The directive classes control such aspects as authorization, client access, and directory indexing. You can specify the argument `All` to permit all classes of directives in `.htaccess` files, a space-separated list of directive classes to permit only those classes, or `None` to make the server ignore `.htaccess` files altogether.

> **Note**
>
> If SELinux is enabled on the system, you must change the default file type if the file system hierarchy specified by `<Directory>` is not under `/var/www/html`.

`<IfModule [!]module>` Applies directives if the specified module has been loaded, or, when the exclamation point (`!`) is specified, if the module has not been loaded.

The following example disallows user-published content if `mod_userdir.c` has been loaded:

```
<IfModule mod_userdir.c>
  UserDir disabled
</IfModule>
```

`<Limit method ...>` Places limits on the specified HTTP methods (such as GET, OPTIONS, POST, and PUT) for use with a Uniform Resource Identifier (URI).

The following example limits systems in `mydom.com` to using only the `GET` and `PUT` methods to perform HTTP downloads and uploads:

```
<Limit GET PUT>
  Order deny,allow
  Deny from all
  Allow from .example.com
</Limit>
```

Systems outside `mydom.com` cannot use `GET` and `PUT` with the URI.

`<LimitExcept method ...>` Places limits on all except the specified HTTP methods for use with a Uniform Resource Identifier (URI).

The following example disallows any system from using any method other than `GET` and `POST`:

```
<LimitExcept GET POST>
  Order deny,allow
  Deny from all
</Limit>
```

`VirtualHost IP_address:port ...` Specifies a group of directives that define a container for a virtual host. See Section 15.6, "Configuring Apache Virtual Hosts".

15.5.1 About Nested Containers

The following example illustrates how you can nest containers, using `<Limit>` and `<LimitExcept>` containers to permit `GET`, `POST`, and `OPTIONS` to be used with user directories under `/home/*/public_html`.

```
<Directory /home/*/public_html>
```

```
  AllowOverride FileInfo AuthConfig Limit
  Options MultiViews Indexes SymLinksIfOwnerMatch \
  IncludesNoExec
  <Limit GET POST OPTIONS>
    Order allow,deny
    Allow from all
  </Limit>
  <LimitExcept GET POST OPTIONS>
    Order deny,allow
    Deny from all
  </LimitExcept>
</Directory>
```

In the example, the `AllowOverride` directive specifies the following directive classes:

`AuthConfig` Permits the use of the authorization directives.

`FileInfo`   Permits the use of directives that control document types.

`Limit`   Permits the use of directives that control host access.

The `Options` directive controls the features of the server for the directory hierarchy, for example:

`FollowSymLinks`   Follow symbolic links under the directory hierarchy.

`Includes`   Permits server-side includes.

`IncludesNoExec`   Prevents the server from running `#exec cmd` and `#exec cgi` server-side
                       includes.

`Indexes`   Generates a web directory listing if the `DirectoryIndex` directive is not set.

`MultiViews`   Allows the server to determine the file to use that best matches the client's
                       requirements based on the MIME type when several versions of the file exist
                       with different extensions.

`SymLinksIfOwnerMatch` Allows the server to follow a symbolic link if the file or directory being pointed
                       to has the same owner as the symbolic link.

For more information, see http://httpd.apache.org/docs/current/mod/directives.html.

15.6 Configuring Apache Virtual Hosts

The Apache HTTP server supports virtual hosts, meaning that it can respond to requests that are directed to multiple IP addresses or host names that correspond to the same host machine. You can configure each virtual host to provide different content and to behave differently.

You can configure virtual hosts in two ways:

• *IP-based Virtual Hosts (host-by-IP)*

  Each virtual host has its own combination of IP address and port. The server responds to the IP address with which the host name resolves. Host-by-IP is needed to server HTTPS requests because of restrictions in the SSL (Secure Sockets Layer) protocol.

• *Name-based Virtual Hosts (host-by-name)*

  All virtual hosts share a common IP address. Apache responds to the request by mapping the host name in the request to `ServerName` and `ServerAlias` directives for the virtual host in the configuration file.

To configure a virtual host, you use the `<VirtualHost hostname>` container. You must also divide all served content between the virtual hosts that you configure.

The following example shows a simple name-based configuration for two virtual hosts:

```
NameVirtualHost *:80

<VirtualHost *:80>
  ServerName websvr1.mydom.com
  ServerAlias www.mydom-1.com
  DocumentRoot /var/www/http/websvr1
  ErrorLog websvr1.error_log
</VirtualHost>

<VirtualHost *:80>
  ServerName websvr2.mydom.com
  ServerAlias www.mydom-2.com
  DocumentRoot /var/www/http/sebsvr2
  ErrorLog websvr2.error_log
</VirtualHost>
```

For more information, see http://httpd.apache.org/docs/2.2/vhosts/.

Chapter 16 Email Service Configuration

Table of Contents

This chapter describes email programs and protocols that are available with Oracle Linux, and how to set up a basic Sendmail client.

16.1 About Email Programs

A Mail User Agent is an email client application that allows you to create and read email messages, set up mailboxes to store and organize messages, and send outbound messages to a Mail Transfer Agent (MTA). Many MUAs can also retrieve email messages from remote servers using the Post Office Protocol (POP) or Internet Message Access Protocol (IMAP).

A Mail Transfer Agent (MTA) transports email messages between systems by using the Simple Mail Transport Protocol (SMTP). The mail delivery services from the client program to a destination server possibly traverses several MTAs in its route. Oracle Linux offers two MTAs, Postfix and Sendmail, and also includes the special purpose MTA, Fetchmail for use with SLIP and PPP.

A Mail Delivery Agent (MDA) performs the actual delivery of an email message. The MTA invokes an MDA, such as Procmail, to place incoming email in the recipient's mailbox file. MDAs distribute and sort messages on the local system that email client application can access.

16.2 About Email Protocols

Several different network protocols are required to deliver email messages. These protocols work together to allow different systems, often running different operating systems and different email programs, to send, transfer, and receive email.

16.2.1 About SMTP

The Simple Mail Transfer Protocol (SMTP) is a transport protocol that provides mail delivery services between email client applications and servers, and between the originating server and the destination server. You must specify the SMTP server when you configure outgoing email for an email client application.

SMTP does not require authentication. Anyone can use SMTP to send email, including junk email and unsolicited bulk email. If you administer an SMTP server, you can configure relay restrictions that limit users from sending email through it. Open relay servers do not have any such restrictions. Both Postfix and Sendmail are SMTP server programs that use SMTP. Unless you own a domain in which you want to receive email, you do not need to set up an SMTP server.

16.2.2 About POP and IMAP

The Post Office Protocol (POP) is an email access protocol that email client applications use to retrieve email messages from the mailbox on a remote server, typically maintained by an Internet Service Provider (ISP). POP email clients usually delete the message on the server when it has been successfully retrieved or within a short time period thereafter.

The Internet Message Access Protocol (IMAP) is an email access protocol that email client applications use to retrieve email messages from a remote server, typically maintained by their organization. The entire message is downloaded only when you open it, and you can delete messages from the server without first downloading them. Email is retained on the server when using IMAP.

Both POP and IMAP allow you to manage mail folders and create multiple mail directories to organize and store email.

The `dovecot` package provides the `dovecot` service that implements both an IMAP server and a POP server.

By default, the dovecot service runs IMAP and POP together with their secure versions that use Secure Socket Layer (SSL) encryption for client authentication and data transfer sessions. The IMAP and POP servers provided by `dovecot` are configured to work as installed. It is usually unnecessary to modify the configuration file, `/etc/dovecot.conf`.

For more information, see the `dovecot(1)` manual page and `/usr/share/doc/dovecot-version`.

16.3 About the Postfix SMTP Server

Postfix is configured as the default MTA on Oracle Linux. Although Postfix does not have as many features as Sendmail, it is easier to administer than Sendmail and its features are sufficient to meet the requirements of most installations. You should only use Sendmail if you want to use address re-writing rules or mail filters (*milters*) that are specific to Sendmail. Most mail filters function correctly with Postfix. If you do use Sendmail, disable or uninstall Postfix to avoid contention over network port usage.

Postfix has a modular design that consists of a master daemon and several smaller processes. Postfix stores its configuration files in the `/etc/postfix` directory, including:

`access`   Specifies which hosts are allowed to connect to Postfix.

`main.cf`   Contains global configuration options for Postfix.

`master.cf`   Specifies how the Postfix master daemon and other Postfix processes interact to deliver email.

`transport`   Specifies the mapping between destination email addresses and relay hosts.

By default, Postfix does not accept network connections from any system other than the local host. To enable mail delivery for other hosts, edit `/etc/postfix/main.cf` and configure their domain, host name, and network information.

Restart the Postfix service after making any configuration changes:

```
# systemctl restart postfix
```

For more information, see `postfix(1)` and other Postfix manual pages, Section 16.5, "Forwarding Email", `/usr/share/doc/postfix-version`, and http://www.postfix.org/documentation.html.

16.4 About the Sendmail SMTP Server

Sendmail is highly configurable and is the most commonly used MTA on the Internet. Sendmail is mainly used to transfer email between systems, but it is capable of controlling almost every aspect of how email is handled.

Sendmail is distributed in the following packages:

procmail   Contains Procmail, which acts as the default local MDA for Sendmail. This package is installed as a dependency of the sendmail package.

sendmail   Contains the Sendmail MTA.

sendmail-cf Contains configuration files for Sendmail.

To install the Sendmail packages, enter:

```
# yum install sendmail sendmail-cf
```

For more information, see the sendmail(8) manual page.

16.4.1 About Sendmail Configuration Files

The main configuration file for Sendmail is /etc/mail/sendmail.cf, which is not intended to be manually edited. Instead, make any configuration changes in the /etc/mail/sendmail.mc file.

If you want Sendmail to relay email from other systems, change the following line in sendmail.mc:

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

so that it reads:

```
dnl # DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

The leading dnl stands for *delete to new line*, and effectively comments out the line.

After you have edited sendmail.mc, restart the sendmail service to regenerate sendmail.cf:

```
# systemctl restart sendmail
```

Alternatively, you can use the make script in /etc/mail:

```
# /etc/mail/make all
```

However, Sendmail does not use the regenerated configuration file until you restart the server.

Other important Sendmail configuration files in /etc/mail include:

access   Configures a relay host that processes outbound mail from the local host to other systems. This is the default configuration:

```
Connect: localhost.localdomain      RELAY
Connect: localhost                   RELAY
Connect: 127.0.0.1                   RELAY
```

To configure Sendmail to relay mail from other systems on a local network, add an entry such as the following:

```
Connect: 192.168.2                        RELAY
```

155

mailertable	Configures forwarding of email from one domain to another. The following example forwards email sent to the yourorg.org domain to the SMTP server for the mydom.com domain:

```
yourorg.org          smtp:[mydom.com]
```

virtusertable	Configures serving of email to multiple domains. Each line starts with a destination address followed by the address to which Sendmail forwards the email. For example, the following entry forwards email addressed to any user at yourorg.org to the same user name at mydom.com:

```
@yourorg.org          %1@mydom.com
```

Each of these configuration files has a corresponding database (.db) file in /etc/mail that Sendmail reads. After making any changes to any of the configuration files, restart the sendmail service. To regenerate the database files, run the /etc/mail/make all command. As for sendmail.cf, Sendmail does not use the regenerated database files until you restart the server.

16.5 Forwarding Email

You can forward incoming email messages with the Postfix local delivery agent or with Sendmail by configuring the /etc/aliases file. Entries in this file can map inbound addresses to local users, files, commands, and remote addresses.

The following example redirects email for postmaster to root, and forwards email sent to admin on the local system to several other users, including usr04, who is on a different system:

```
postmaster:    root
admin:         usr01, usr02, usr03, usr04@another-system.com
```

To direct email to a file, specify an absolute path name instead of the destination address. To specify a command, precede it with a pipe character (|). The next example erases email sent to nemo by sending it to /dev/null, and runs a script named aggregator to process emails sent to fixme:

```
nemo:          /dev/null
fixme:         |/usr/local/bin/aggregator
```

After changing the file, run the command newaliases to rebuild the indexed database file.

For more information, see the aliases(5) manual page.

16.6 Configuring a Sendmail Client

A Sendmail client sends outbound mail to another SMTP server, which is typically administered by an ISP or the IT department of an organization, and this server then relays the email to its destination.

To configure a Sendmail client:

1. If the account on the SMTP server requires authentication:

   a. Create an auth directory under /etc/mail that is accessible only to root:

      ```
      # mkdir /etc/mail/auth
      # chmod 700 /etc/mail/auth
      ```

   b. In the auth directory, create a file smtp-auth that contains the authentication information for the SMTP server, for example:

```
# echo 'AuthInfo:smtp.isp.com: "U:username" "P:password"' > /etc/mail/auth/smtp-auth
```

where `smtp.isp.com` is the FQDN of the SMTP server, and `username` and `password` are the name and password of the account.

c. Create the database file from `smtp-auth`, and make both files read-writable only by `root`:

```
# cd /etc/mail/auth
# makemap hash smtp-auth < smtp-auth
# chmod 600 smtp-auth smtp-auth.db
```

2. Edit `/etc/mail/sendmail.mc`, and change the following line:

```
dnl define('SMART_host', 'smtp.your.provider')dnl
```

to read:

```
define('SMART_host', 'smtp.isp.com')dnl
```

where `smtp.isp.com` is the FQDN of the SMTP server.

3. If the account on the SMTP server requires authentication, add the following lines after the line that defines `SMART_host`:

```
define('RELAY_MAILER_ARGS', 'TCP $h port')dnl
define('confAUTH_MECHANISMS', 'EXTERNAL GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
FEATURE('authinfo','hash /etc/mail/auth/smtp-auth.db')dnl
define(`confAUTH_OPTIONS', `A p y')dnl
```

where `port` is the port number used by the SMTP server (for example, 587 for SMARTTLS or 465 for SSL/TLS).

4. Edit `/etc/sysconfig/sendmail` and set the value of `DAEMON` to `no`:

```
DAEMON=no
```

This entry disables `sendmail` from listening on port 25 for incoming email.

5. Restart the `sendmail` service:

```
# systemctl restart sendmail
```

To test the configuration, send email to an account in another domain.

This configuration does not receive or relay incoming email. You can use a client application to receive email via POP or IMAP.

Chapter 17 Load Balancing and High Availability Configuration

Table of Contents

This chapter describes how to configure the Keepalived and HAProxy technologies for balancing access to network services while maintaining continuous access to those services.

17.1 About HAProxy

HAProxy is an application layer (Layer 7) load balancing and high availability solution that you can use to implement a reverse proxy for HTTP and TCP-based Internet services.

The configuration file for the `haproxy` daemon is `/etc/haproxy/haproxy.cfg`. This file must be present on each server on which you configure HAProxy for load balancing or high availability.

For more information, see http://www.haproxy.org/#docs, the `/usr/share/doc/haproxy-version` documentation, and the `haproxy(1)` manual page.

17.2 Installing and Configuring HAProxy

To install HAProxy:

1. Install the `haproxy` package on each front-end server:

   ```
   # yum install haproxy
   ```

2. Edit `/etc/haproxy/haproxy.cfg` to configure HAProxy on each server. See Section 17.2.1, "About the HAProxy Configuration File".

3. Enable IP forwarding and binding to non-local IP addresses:

   ```
   # echo "net.ipv4.ip_forward = 1" >> /etc/sysctl.conf
   # echo "net.ipv4.ip_nonlocal_bind = 1" >> /etc/sysctl.conf
   # sysctl -p
   ```

```
net.ipv4.ip_forward = 1
net.ipv4.ip_nonlocal_bind = 1
```

4. Enable access to the services or ports that you want HAProxy to handle.

   For example, to enable access to HTTP and make this rule persist across reboots, enter the following commands:

   ```
   # firewall-cmd --zone=zone --add-service=http
   success
   # firewall-cmd --permanent --zone=zone --add-service=http
   success
   ```

   To allow incoming TCP requests on port 8080:

   ```
   # firewall-cmd --zone=zone --add-port=8080/tcp
   success
   # firewall-cmd --permanent --zone=zone --add-port=8080/tcp
   success
   ```

5. Enable and start the haproxy service on each server:

   ```
   # systemctl enable haproxy
   ln -s '/usr/lib/systemd/system/haproxy.service' \
      '/etc/systemd/system/multi-user.target.wants/haproxy.service'
   # systemctl start haproxy
   ```

   If you change the HAProxy configuration, reload the haproxy service:

   ```
   # systemctl reload haproxy
   ```

17.2.1 About the HAProxy Configuration File

The /etc/haproxy/haproxy.cfg configuration file is divided into the following sections:

global	Defines global settings such as the syslog facility and level to use for logging, the maximum number of concurrent connections allowed, and how many processes to start in daemon mode.
defaults	Defines default settings for subsequent sections.
listen	Defines a complete proxy, implicitly including the frontend and backend components.
frontend	Defines the ports that accept client connections.
backend	Defines the servers to which the proxy forwards client connections.

For examples of how to configure HAProxy, see:

- Section 17.3, "Configuring Simple Load Balancing Using HAProxy"

- Section 17.10, "Making HAProxy Highly Available Using Keepalived"

- Section 17.12, "Making HAProxy Highly Available Using Oracle Clusterware"

17.3 Configuring Simple Load Balancing Using HAProxy

The following example uses HAProxy to implement a front-end server that balances incoming requests between two back-end web servers, and which is also able to handle service outages on the back-end servers.

Figure 17.1 shows an HAProxy server (10.0.0.10), which is connected to an externally facing network (10.0.0/24) and to an internal network (192.168.1/24). Two web servers, `websvr1` (192.168.1.71) and `websvr2` (192.168.1.72), are accessible on the internal network. The IP address 10.0.0.10 is in the private address range 10.0.0/24, which cannot be routed on the Internet. An upstream network address translation (NAT) gateway or a proxy server provides access to and from the Internet.

Figure 17.1 Example HAProxy Configuration for Load Balancing
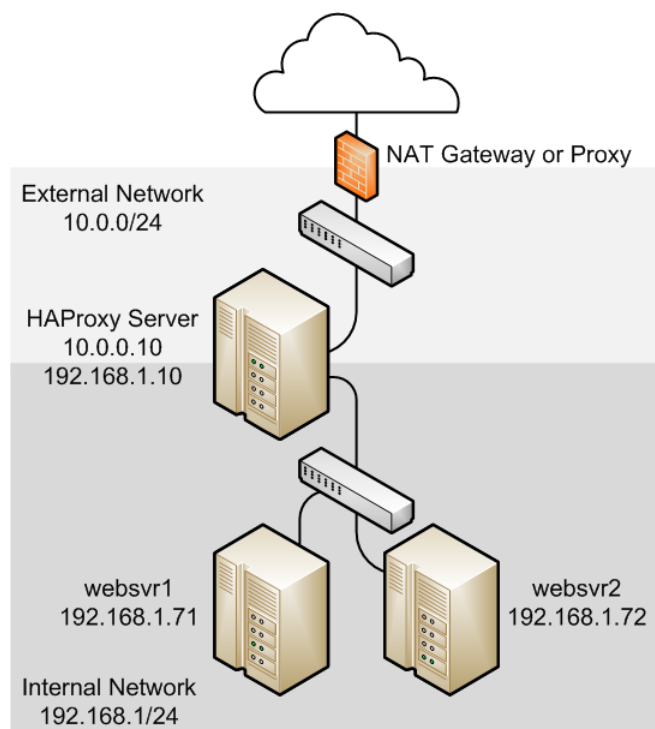
You might use the following configuration in `/etc/haproxy/haproxy.cfg` on the server:

```
global
    daemon
    log 127.0.0.1 local0 debug
    maxconn 50000
    nbproc 1

defaults
    mode http
    timeout connect 5s
    timeout client 25s
    timeout server 25s
    timeout queue 10s

# Handle Incoming HTTP Connection Requests
listen  http-incoming
    mode http
    bind 10.0.0.10:80
# Use each server in turn, according to its weight value
    balance roundrobin
# Verify that service is available
    option httpchk OPTIONS * HTTP/1.1\r\nHost:\ www
# Insert X-Forwarded-For header
    option forwardfor
# Define the back-end servers, which can handle up to 512 concurrent connections each
    server websvr1 192.168.1.71:80 weight 1 maxconn 512 check
    server websvr2 192.168.1.72:80 weight 1 maxconn 512 check
```

This configuration balances HTTP traffic between the two back-end web servers `websvr1` and `websvr2`, whose firewalls are configured to accept incoming TCP requests on port 80.

After implementing simple `/var/www/html/index.html` files on the web servers and using `curl` to test connectivity, the following output demonstrate how HAProxy balances the traffic between the servers and how it handles the `httpd` service stopping on `websvr1`:

```
$ while true; do curl http://10.0.0.10; sleep 1; done
This is HTTP server websvr1 (192.168.1.71).
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr1 (192.168.1.71).
This is HTTP server websvr2 (192.168.1.72).
...
This is HTTP server websvr2 (192.168.1.72).
<html><body><h1>503 Service Unavailable</h1>
No server is available to handle this request.
</body></html>
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr2 (192.168.1.72).
...
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr1 (192.168.1.71).
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr1 (192.168.1.71).
...
^C
$
```

In this example, HAProxy detected that the `httpd` service had restarted on `websvr1` and resumed using that server in addition to `websvr2`.

By combining the load balancing capability of HAProxy with the high availability capability of Keepalived or Oracle Clusterware, you can configure a backup load balancer that ensures continuity of service in the event that the master load balancer fails. See Section 17.10, "Making HAProxy Highly Available Using Keepalived" and Section 17.12, "Making HAProxy Highly Available Using Oracle Clusterware".

See Section 17.2, "Installing and Configuring HAProxy" for details of how to install and configure HAProxy.

17.3.1 Configuring HAProxy for Session Persistence

Many web-based application require that a user session is persistently served by the same web server.

If you want web sessions to have persistent connections to the same server, you can use a `balance` algorithm such as `hdr`, `rdp-cookie`, `source`, `uri`, or `url_param`.

If your implementation requires the use of the `leastconn`, `roundrobin`, or `static-rr` algorithm, you can implement session persistence by using server-dependent cookies.

To enable session persistence for all pages on a web server, use the `cookie` directive to define the name of the cookie to be inserted and add the `cookie` option and server name to the `server` lines, for example:

```
    cookie WEBSVR insert
    server websvr1 192.168.1.71:80 weight 1 maxconn 512 cookie 1 check
    server websvr2 192.168.1.72:80 weight 1 maxconn 512 cookie 2 check
```

HAProxy includes an additional `Set-Cookie:` header that identifies the web server in its response to the client, for example: `Set-Cookie: WEBSVR=N; path=page_path`. If a client subsequently

specifies the `WEBSVR` cookie in a request, HAProxy forwards the request to the web server whose `server cookie`value matches the value of `WEBSVR`.

The following example demonstrates how an inserted cookie ensures session persistence:

```
$ while true; do curl http://10.0.0.10; sleep 1; done
This is HTTP server websvr1 (192.168.1.71).
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr1 (192.168.1.71).
^C
$ curl http://10.0.0.10 -D /dev/stdout
HTTP/1.1 200 OK
Date: ...
Server: Apache/2.4.6 ()
Last-Modified: ...
ETag: "26-5125afd089491"
Accept-Ranges: bytes
Content-Length: 38
Content-Type: text/html; charset=UTF-8
Set-Cookie: WEBSVR=2; path=/

This is HTTP server svr2 (192.168.1.72).
$ while true; do curl http://10.0.0.10 --cookie "WEBSVR=2;"; sleep 1; done
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr2 (192.168.1.72).
This is HTTP server websvr2 (192.168.1.72).
^C
```

To enable persistence selectively on a web server, use the `cookie` directive to specify that HAProxy should expect the specified cookie, usually a session ID cookie or other existing cookie, to be prefixed with the `server cookie` value and a `~` delimiter, for example:

```
cookie SESSIONID prefix
server websvr1 192.168.1.71:80 weight 1 maxconn 512 cookie 1 check
server websvr2 192.168.1.72:80 weight 1 maxconn 512 cookie 2 check
```

If the value of `SESSIONID` is prefixed with a `server cookie` value, for example: `Set-Cookie: SESSIONID=N~Session_ID;`, HAProxy strips the prefix and delimiter from the `SESSIONID` cookie before forwarding the request to the web server whose `server cookie` value matches the prefix.

The following example demonstrates how using a prefixed cookie enables session persistence:

```
$ while true; do curl http://10.0.0.10 --cookie "SESSIONID=1~1234;"; sleep 1; done
This is HTTP server websvr1 (192.168.1.71).
This is HTTP server websvr1 (192.168.1.71).
This is HTTP server websvr1 (192.168.1.71).
^C
```

A real web application would usually set the session ID on the server side, in which case the first HAProxy response would include the prefixed cookie in the `Set-Cookie:` header.

17.4 About Keepalived

Keepalived uses the IP Virtual Server (IPVS) kernel module to provide transport layer (Layer 4) load balancing, redirecting requests for network-based services to individual members of a server cluster. IPVS monitors the status of each server and uses the Virtual Router Redundancy Protocol (VRRP) to implement high availability.

The configuration file for the `keepalived` daemon is `/etc/keepalived/keepalived.conf`. This file must be present on each server on which you configure Keepalived for load balancing or high availability.

For more information, see http://www.keepalived.org/documentation.html, the `/usr/share/doc/keepalive-version` documentation, and the `keepalived(8)` and `keepalived.conf(5)` manual pages.

17.5 Installing and Configuring Keepalived

To install Keepalived:

1. Install the `keepalived` package on each server:

```
# yum install keepalived
```

2. Edit `/etc/keepalived/keepalived.conf` to configure Keepalived on each server. See Section 17.5.1, "About the Keepalived Configuration File".

3. Enable IP forwarding:

```
# echo "net.ipv4.ip_forward = 1" >> /etc/sysctl.conf
# sysctl -p
net.ipv4.ip_forward = 1
```

4. Add firewall rules to allow VRRP communication using the multicast IP address 224.0.0.18 and the VRRP protocol (112) on each network interface that Keepalived will control, for example:

```
# firewall-cmd --direct --permanent --add-rule ipv4 filter INPUT 0 \
  --in-interface enp0s8 --destination 224.0.0.18 --protocol vrrp -j ACCEPT
success
# firewall-cmd --direct --permanent --add-rule ipv4 filter OUTPUT 0 \
  --out-interface enp0s8 --destination 224.0.0.18 --protocol vrrp -j ACCEPT
success
# firewall-cmd --reload
success
```

5. Enable and start the `keepalived` service on each server:

```
# systemctl enable keepalived
ln -s '/usr/lib/systemd/system/keepalived.service' \
  '/etc/systemd/system/multi-user.target.wants/keepalived.service'
# systemctl start keepalived
```

If you change the Keepalived configuration, reload the `keepalived` service:

```
# systemctl reload keepalived
```

17.5.1 About the Keepalived Configuration File

The `/etc/keepalived/keepalived.conf` configuration file is divided into the following sections:

`global_defs`	Defines global settings such as the email addresses for sending notification messages, the IP address of an SMTP server, the timeout value for SMTP connections in seconds, a string that identifies the host machine, the VRRP IPv4 and IPv6 multicast addresses, and whether SNMP traps should be enabled.
`static_ipaddress`, `static_routes`	Define static IP addresses and routes, which VRRP cannot change. These sections are not required if the addresses and routes are already defined on the servers and these servers already have network connectivity.
`vrrp_sync_group`	Defines a VRRP synchronization group of VRRP instances that fail over together.

`vrrp_instance`	Defines a moveable virtual IP address for a member of a VRRP synchronization group's internal or external network interface, which accompanies other group members during a state transition. Each VRRP instance must have a unique value of `virtual_router_id`, which identifies which interfaces on the master and backup servers can be assigned a given virtual IP address. You can also specify scripts that are run on state transitions to `BACKUP`, `MASTER`, and `FAULT`, and whether to trigger SMTP alerts for state transitions.
`vrrp_script`	Defines a tracking script that Keepalived can run at regular intervals to perform monitoring actions from a `vrrp_instance` or `vrrp_sync_group` section.
`virtual_server_group`	Defines a virtual server group, which allows a real server to be a member of several virtual server groups.
`virtual_server`	Defines a virtual server for load balancing, which is composed of several real servers.

For examples of how to configure Keepalived, see:

- Section 17.6, "Configuring Simple Virtual IP Address Failover Using Keepalived"

- Section 17.7, "Configuring Load Balancing Using Keepalived in NAT Mode"

- Section 17.8, "Configuring Load Balancing Using Keepalived in DR Mode"

- Section 17.10, "Making HAProxy Highly Available Using Keepalived"

17.6 Configuring Simple Virtual IP Address Failover Using Keepalived

A typical Keepalived high-availability configuration consists of one master server and one or more backup servers. One or more virtual IP addresses, defined as *VRRP instances*, are assigned to the master server's network interfaces so that it can service network clients. The backup servers listen for multicast VRRP advertisement packets that the master server transmits at regular intervals. The default advertisement interval is one second. If the backup nodes fail to receive three consecutive VRRP advertisements, the backup server with the highest assigned priority takes over as the master server and assigns the virtual IP addresses to its own network interfaces. If several backup servers have the same priority, the backup server with the highest IP address value becomes the master server.

The following example uses Keepalived to implement a simple failover configuration on two servers. One server acts as the master, the other acts as a backup, and the master server has a higher priority than the backup server.

Figure 17.2 shows how the virtual IP address 10.0.0.100 is initially assigned to the master server (10.0.0.71). When the master server fails, the backup server (10.0.0.72) becomes the new master server and is assigned the virtual IP address 10.0.0.100.

Figure 17.2 Example Keepalived Configuration for Virtual IP Address Failover

You might use the following configuration in `/etc/keepalived/keepalived.conf` on the master server:

```
global_defs {
   notification_email {
     root@mydomain.com
   }
   notification_email_from svr1@mydomain.com
   smtp_server localhost
   smtp_connect_timeout 30
}

vrrp_instance VRRP1 {
    state MASTER
#   Specify the network interface to which the virtual address is assigned
    interface enp0s8
#   The virtual router ID must be unique to each VRRP instance that you define
    virtual_router_id 41
#   Set the value of priority higher on the master server than on a backup server
    priority 200
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1066
    }
    virtual_ipaddress {
        10.0.0.100/24
    }
}
```

The configuration of the backup server is the same except for the values of `notification_email_from`, `state`, `priority`, and possibly `interface` if the system hardware configuration is different:

```
global_defs {
   notification_email {
     root@mydomain.com
   }
   notification_email_from svr2@mydomain.com
   smtp_server localhost
```

```
    smtp_connect_timeout 30
}

vrrp_instance VRRP1 {
    state BACKUP
#   Specify the network interface to which the virtual address is assigned
    interface enp0s8
    virtual_router_id 41
#   Set the value of priority lower on the backup server than on the master server
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1066
    }
    virtual_ipaddress {
        10.0.0.100/24
    }
}
```

In the event that the master server (`svr1`) fails, `keepalived` assigns the virtual IP address 10.0.0.100/24 to the `enp0s8` interface on the backup server (`svr2`), which becomes the master server.

To determine whether a server is acting as the master, you can use the `ip` command to see whether the virtual address is active, for example:

```
# ip addr list enp0s8
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:cb:a6:8d brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.72/24 brd 10.0.0.255 scope global enp0s8
    inet 10.0.0.100/24 scope global enp0s8
    inet6 fe80::a00:27ff:fecb:a68d/64 scope link
       valid_lft forever preferred_lft forever
```

Alternatively, search for Keepalived messages in `/var/log/messages` that show transitions between states, for example:

```
...51:55 ... VRRP_Instance(VRRP1) Entering BACKUP STATE
...
...53:08 ... VRRP_Instance(VRRP1) Transition to MASTER STATE
...53:09 ... VRRP_Instance(VRRP1) Entering MASTER STATE
...53:09 ... VRRP_Instance(VRRP1) setting protocol VIPs.
...53:09 ... VRRP_Instance(VRRP1) Sending gratuitous ARPs on enp0s8 for 10.0.0.100
```

Note

Only one server should be active as the master at any time. If more than one server is configured as the master, it is likely that there is a problem with VRRP communication between the servers. Check the network settings for each interface on each server and check that the firewall allows both incoming and outgoing VRRP packets for multicast IP address 224.0.0.18.

See Section 17.5, "Installing and Configuring Keepalived" for details of how to install and configure Keepalived.

17.7 Configuring Load Balancing Using Keepalived in NAT Mode

The following example uses Keepalived in NAT mode to implement a simple failover and load balancing configuration on two servers. One server acts as the master, the other acts as a backup, and the master server has a higher priority than the backup server. Each of the servers has two network interfaces, where one interface is connected to the side facing an external network (192.168.1.0/24) and the other interface is connected to an internal network (10.0.0.0/24) on which two web servers are accessible.

Figure 17.3 shows that the Keepalived master server has network addresses 192.168.1.10, 192.168.1.1 (virtual), 10.0.0.10, and 10.0.0.100 (virtual). The Keepalived backup server has network addresses 192.168.1.11 and 10.0.0.11. The web servers `websvr1` and `websvr2` have network addresses 10.0.0.71 and 10.0.0.72 respectively.

Figure 17.3 Example Keepalived Configuration for Load Balancing in NAT Mode

You might use the following configuration in `/etc/keepalived/keepalived.conf` on the master server:

```
global_defs {
   notification_email {
     root@mydomain.com
   }
   notification_email_from svr1@mydomain.com
   smtp_server localhost
   smtp_connect_timeout 30
}

vrrp_sync_group VRRP1 {
#   Group the external and internal VRRP instances so they fail over together
    group {
        external
        internal
        }
}

vrrp_instance external {
    state MASTER
    interface enp0s8
    virtual_router_id 91
    priority 200
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1215
    }
```

```
#   Define the virtual IP address for the external network interface
    virtual_ipaddress {
        192.168.1.1/24
    }
}

vrrp_instance internal {
    state MASTER
    interface enp0s9
    virtual_router_id 92
    priority 200
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1215
    }
#   Define the virtual IP address for the internal network interface
    virtual_ipaddress {
        10.0.0.100/24
    }
}

# Define a virtual HTTP server on the virtual IP address 192.168.1.1
virtual_server 192.168.1.1 80 {
    delay_loop 10
    protocol TCP
#   Use round-robin scheduling in this example
    lb_algo rr
#   Use NAT to hide the back-end servers
    lb_kind NAT
#   Persistence of client sessions times out after 2 hours
    persistence_timeout 7200

    real_server 10.0.0.71 80 {
        weight 1
        TCP_CHECK {
          connect_timeout 5
          connect_port 80
        }
    }

    real_server 10.0.0.72 80 {
        weight 1
        TCP_CHECK {
          connect_timeout 5
          connect_port 80
        }
    }
}
```

This configuration is similar to that given in Section 17.6, "Configuring Simple Virtual IP Address Failover Using Keepalived" with the additional definition of a `vrrp_sync_group` section so that the network interfaces are assigned together on failover, and a `virtual_server` section to define the real back-end servers that Keepalived uses for load balancing. The value of `lb_kind` is set to `NAT` (Network Address Translation), which means that the Keepalived server handles both inbound and outbound network traffic from and to the client on behalf of the back-end servers.

The configuration of the backup server is the same except for the values of `notification_email_from`, `state`, `priority`, and possibly `interface` if the system hardware configuration is different:

```
global_defs {
   notification_email {
     root@mydomain.com
   }
```

```
    notification_email_from svr2@mydomain.com
    smtp_server localhost
    smtp_connect_timeout 30
}

vrrp_sync_group VRRP1 {
#   Group the external and internal VRRP instances so they fail over together
    group {
        external
        internal
        }
}

vrrp_instance external {
    state BACKUP
    interface enp0s8
    virtual_router_id 91
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1215
    }
#   Define the virtual IP address for the external network interface
    virtual_ipaddress {
        192.168.1.1/24
    }
}

vrrp_instance internal {
    state BACKUP
    interface enp0s9
    virtual_router_id 92
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1215
    }
#   Define the virtual IP address for the internal network interface
    virtual_ipaddress {
        10.0.0.100/24
    }
}

# Define a virtual HTTP server on the virtual IP address 192.168.1.1
virtual_server 192.168.1.1 80 {
    delay_loop 10
    protocol TCP
#   Use round-robin scheduling in this example
    lb_algo rr
#   Use NAT to hide the back-end servers
    lb_kind NAT
#   Persistence of client sessions times out after 2 hours
    persistence_timeout 7200

    real_server 10.0.0.71 80 {
        weight 1
        TCP_CHECK {
          connect_timeout 5
          connect_port 80
        }
    }

    real_server 10.0.0.72 80 {
        weight 1
        TCP_CHECK {
```

```
        connect_timeout 5
        connect_port 80
      }
    }
}
```

Two further configuration changes are required:

- Configure firewall rules on each Keepalived server (master and backup) that you configure as a load balancer as described in Section 17.7.1, "Configuring Firewall Rules for Keepalived NAT-Mode Load Balancing".

- Configure a default route for the virtual IP address of the load balancer's internal network interface on each back-end server that you intend to use with the Keepalived load balancer as described in Section 17.7.2, "Configuring Back-End Server Routing for Keepalived NAT-Mode Load Balancing".

See Section 17.5, "Installing and Configuring Keepalived" for details of how to install and configure Keepalived.

17.7.1 Configuring Firewall Rules for Keepalived NAT-Mode Load Balancing

If you configure Keepalived to use NAT mode for load balancing with the servers on the internal network, the Keepalived server handles all inbound and outbound network traffic and hides the existence of the back-end servers by rewriting the source IP address of the real back-end server in outgoing packets with the virtual IP address of the external network interface.

To configure a Keepalived server to use NAT mode for load balancing:

1. Configure the firewall so that the interfaces on the external network side are in a different zone from the interfaces on the internal network side.

   The following example demonstrates how to move interface `enp0s9` to the `internal` zone while interface `enp0s8` remains in the `public` zone:

   ```
   # firewall-cmd --get-active-zones
   public
     interfaces: enp0s8 enp0s9
   # firewall-cmd --zone=public --remove-interface=enp0s9
   success
   # firewall-cmd --zone=internal --add-interface=enp0s9
   success
   # firewall-cmd --permanent --zone=public --remove-interface=enp0s9
   success
   # firewall-cmd --permanent --zone=internal --add-interface=enp0s9
   success
   # firewall-cmd --get-active-zones
   internal
     interfaces: enp0s9
   public
     interfaces: enp0s8
   ```

2. Configure NAT mode (masquerading) on the external network interface, for example:

   ```
   # firewall-cmd --zone=public --add-masquerade
   success
   # firewall-cmd --permanent --zone=public --add-masquerade
   success
   # firewall-cmd --zone=public --query-masquerade
   yes
   # firewall-cmd --zone=internal --query-masquerade
   no
   ```

3. If not already enabled for your firewall, configure forwarding rules between the external and internal network interfaces, for example:

```
# firewall-cmd --direct --permanent --add-rule ipv4 filter FORWARD 0 \
  -i enp0s8 -o enp0s9 -m state --state RELATED,ESTABLISHED -j ACCEPT
success
# firewall-cmd --direct --permanent --add-rule ipv4 filter FORWARD 0 \
  -i enp0s9 -o enp0s8 -j ACCEPT
success
# firewall-cmd --direct --permanent --add-rule ipv4 filter FORWARD 0 \
  -j REJECT --reject-with icmp-host-prohibited
success
# firewall-cmd --reload
```

4. Enable access to the services or ports that you want Keepalived to handle.

   For example, to enable access to HTTP and make this rule persist across reboots, enter the following commands:

```
# firewall-cmd --zone=public --add-service=http
success
# firewall-cmd --permanent --zone=public --add-service=http
success
```

17.7.2 Configuring Back-End Server Routing for Keepalived NAT-Mode Load Balancing

On each back-end real servers that you intend to use with the Keepalived load balancer, ensure that the routing table contains a default route for the virtual IP address of the load balancer's internal network interface.

For example, if the virtual IP address is `10.0.0.100`, you can use the `ip` command to examine the routing table and to set the default route:

```
# ip route show
10.0.0.0/24 dev enp0s8  proto kernel  scope link  src 10.0.0.71
# ip route add default via 10.0.0.100 dev enp0s8
# ip route show
default via 10.0.0.100 dev enp0s8
10.0.0.0/24 dev enp0s8  proto kernel  scope link  src 10.0.0.71
```

To make the default route for `enp0s8` persist across reboots, create the file `/etc/sysconfig/network-scripts/route-enp0s8`:

```
# echo "default via 10.0.0.100 dev enp0s8" > /etc/sysconfig/network-scripts/route-enp0s8
```

17.8 Configuring Load Balancing Using Keepalived in DR Mode

The following example uses Keepalived in direct routing (DR) mode to implement a simple failover and load balancing configuration on two servers. One server acts as the master, the other acts as a backup, and the master server has a higher priority than the backup server. Each of Keepalived servers has a single network interface and the servers are connected to the same network segment (10.0.0.0/24) on which two web servers are accessible.

Figure 17.4 shows that the Keepalived master server has network addresses 10.0.0.11 and 10.0.0.1 (virtual). The Keepalived backup server has network address 10.0.0.12. The web servers `websvr1` and `websvr2` have network addresses 10.0.0.71 and 10.0.0.72 respectively. In additional, both web servers are configured with the virtual IP address 10.0.0.1 to make them accept packets with that destination address. Incoming requests are received by the master server and redirected to the web servers, which respond directly.

Figure 17.4 Example Keepalived Configuration for Load Balancing in DR Mode

You might use the following configuration in `/etc/keepalived/keepalived.conf` on the master server:

```
global_defs {
   notification_email {
     root@mydomain.com
   }
   notification_email_from svr1@mydomain.com
   smtp_server localhost
   smtp_connect_timeout 30
}

vrrp_instance external {
    state MASTER
    interface enp0s8
    virtual_router_id 91
    priority 200
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1215
    }
    virtual_ipaddress {
        10.0.0.1/24
    }
}

virtual_server 10.0.0.1 80 {
    delay_loop 10
    protocol TCP
    lb_algo rr
#   Use direct routing
    lb_kind DR
    persistence_timeout 7200

    real_server 10.0.0.71 80 {
        weight 1
        TCP_CHECK {
          connect_timeout 5
          connect_port 80
```

```
        }
    }

    real_server 10.0.0.72 80 {
        weight 1
        TCP_CHECK {
          connect_timeout 5
          connect_port 80
        }
    }
}
```

The virtual server configuration is similar to that given in Section 17.7, "Configuring Load Balancing Using Keepalived in NAT Mode" except that the value of `lb_kind` is set to `DR` (Direct Routing), which means that the Keepalived server handles all inbound network traffic from the client before routing it to the back-end servers, which reply directly to the client, bypassing the Keepalived server. This configuration reduces the load on the Keepalived server but is less secure as each back-end server requires external access and is potentially exposed as an attack surface. Some implementations use an additional network interface with a dedicated gateway for each web server to handle the response network traffic.

The configuration of the backup server is the same except for the values of `notification_email_from`, `state`, `priority`, and possibly `interface` if the system hardware configuration is different:

```
global_defs {
   notification_email {
     root@mydomain.com
   }
   notification_email_from svr2@mydomain.com
   smtp_server localhost
   smtp_connect_timeout 30
}

vrrp_instance external {
    state BACKUP
    interface enp0s8
    virtual_router_id 91
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1215
    }
    virtual_ipaddress {
        10.0.0.1/24
    }
}

virtual_server 10.0.0.1 80 {
    delay_loop 10
    protocol TCP
    lb_algo rr
#   Use direct routing
    lb_kind DR
    persistence_timeout 7200

    real_server 10.0.0.71 80 {
        weight 1
        TCP_CHECK {
          connect_timeout 5
          connect_port 80
        }
    }

    real_server 10.0.0.72 80 {
```

```
        weight 1
        TCP_CHECK {
          connect_timeout 5
          connect_port 80
        }
    }
}
```

Two further configuration changes are required:

• Configure firewall rules on each Keepalived server (master and backup) that you configure as a load balancer as described in Section 17.8.1, "Configuring Firewall Rules for Keepalived DR-Mode Load Balancing".

• Configure the `arp_ignore` and `arp_announce` ARP parameters and the virtual IP address for the network interface on each back-end server that you intend to use with the Keepalived load balancer as described in Section 17.8.2, "Configuring the Back-End Servers for Keepalived DR-Mode Load Balancing".

See Section 17.5, "Installing and Configuring Keepalived" for details of how to install and configure Keepalived.

17.8.1 Configuring Firewall Rules for Keepalived DR-Mode Load Balancing

Enable access to the services or ports that you want Keepalived to handle.

For example, to enable access to HTTP and make this rule persist across reboots, enter the following commands:

```
# firewall-cmd --zone=public --add-service=http
success
# firewall-cmd --permanent --zone=public --add-service=http
success
```

17.8.2 Configuring the Back-End Servers for Keepalived DR-Mode Load Balancing

The example configuration requires that the virtual IP address is configured on the master Keepalived server and on each back-end server. The Keepalived configuration maintains the virtual IP address on the master Keepalived server.

Only the master Keepalived server should respond to ARP requests for the virtual IP address. You can set the `arp_ignore` and `arp_announce` ARP parameters for the network interface of each back-end server so that they do not respond to ARP requests for the virtual IP address.

To configure the ARP parameters and virtual IP address on each back-end server:

1. Configure the ARP parameters for the primary network interface, for example `enp0s8`:

```
# echo "net.ipv4.conf.enp0s8.arp_ignore = 1" >> /etc/sysctl.conf
# echo "net.ipv4.conf.enp0s8.arp_announce = 2" >> /etc/sysctl.conf
# sysctl -p
net.ipv4.conf.enp0s8.arp_ignore = 1
net.ipv4.conf.enp0s8.arp_announce = 2
```

2. To define a virtual IP address that persists across reboots, edit `/etc/sysconfig/network-scripts/ifcfg-iface` and add `IPADDR1` and `PREFIX1` entries for the virtual IP address, for example:

```
...
```

```
NAME=enp0s8
...
IPADDR0=10.0.0.72
GATEWAY0=10.0.0.100
PREFIX0=24
IPADDR1=10.0.0.1
PREFIX1=24
...
```

This example defines the virtual IP address 10.0.0.1 for `enp0s8` in addition to the existing real IP address of the back-end server.

3. Reboot the system and verify that the virtual IP address has been set up:

```
# ip addr show enp0s8
2: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:cb:a6:8d brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.72/24 brd 10.0.0.255 scope global enp0s8
    inet 10.0.0.1/24 brd 10.0.0.255 scope global secondary enp0s8
    inet6 fe80::a00:27ff:fecb:a68d/64 scope link
        valid_lft forever preferred_lft forever
```

17.9 Configuring Keepalived for Session Persistence and Firewall Marks

Many web-based application require that a user session is persistently served by the same web server.

If you enable the load balancer in Keepalived to use persistence, a client connects to the same server provided that the timeout period (`persistence_timeout`) has not been exceeded since the previous connection.

Firewall marks are another method for controlling session access so that Keepalived forwards a client's connections on different ports, such as HTTP (80) and HTTPS (443), to the same server, for example:

```
# firewall-cmd --direct --permanent --add-rule ipv4 mangle PREROUTING 0 \
  -d virtual_IP_addr/32 -p tcp -m multiport --dports 80,443 -j MARK --set-mark 123
success
# firewall-cmd --reload
```

These commands set a firewall mark value of 123 on packets that are destined for ports 80 or 443 at the specified virtual IP address.

You must also declare the firewall mark (`fwmark`) value to Keepalived by setting it on the virtual server instead of a destination virtual IP address and port, for example:

```
virtual_server fwmark 123 {
    ...
}
```

This configuration causes Keepalived to route the packets based on their firewall mark value rather than the destination virtual IP address and port. When used in conjunction with session persistence, firewall marks help ensure that all ports used by a client session are handled by the same server.

17.10 Making HAProxy Highly Available Using Keepalived

The following example uses Keepalived to make the HAProxy service fail over to a backup server in the event that the master server fails.

Figure 17.5 shows two HAProxy servers, which are connected to an externally facing network (10.0.0/24) as 10.0.0.11 and 10.0.0.12 and to an internal network (192.168.1/24) as 192.168.1.11 and 192.168.1.12.

One HAProxy server (10.0.0.11) is configured as a Keepalived master server with the virtual IP address 10.0.0.10 and the other (10.0.0.12) is configured as a Keepalived backup server. Two web servers, websvr1 (192.168.1.71) and websvr2 (192.168.1.72), are accessible on the internal network. The IP address 10.0.0.10 is in the private address range 10.0.0/24, which cannot be routed on the Internet. An upstream network address translation (NAT) gateway or a proxy server provides access to and from the Internet.

Figure 17.5 Example of a Combined HAProxy and Keepalived Configuration with Web Servers on a Separate Network
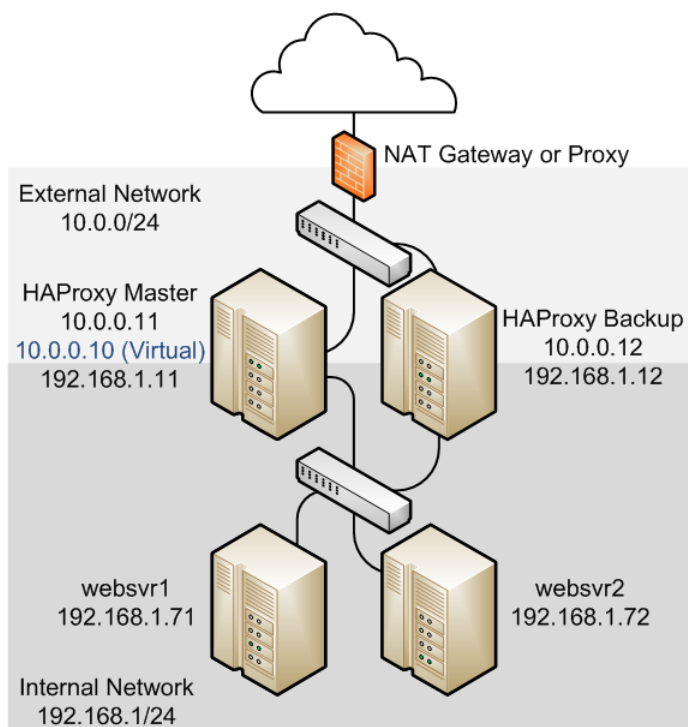
The HAProxy configuration on both 10.0.0.11 and 10.0.0.12 is very similar to Section 17.3, "Configuring Simple Load Balancing Using HAProxy". The IP address on which HAProxy listens for incoming requests is the virtual IP address that Keepalived controls.

```
global
    daemon
    log 127.0.0.1 local0 debug
    maxconn 50000
    nbproc 1

defaults
    mode http
    timeout connect 5s
    timeout client 25s
    timeout server 25s
    timeout queue 10s

# Handle Incoming HTTP Connection Requests on the virtual IP address controlled by Keepalived
listen  http-incoming
    mode http
    bind 10.0.0.10:80
# Use each server in turn, according to its weight value
    balance roundrobin
# Verify that service is available
    option httpchk OPTIONS * HTTP/1.1\r\nHost:\ www
```

```
# Insert X-Forwarded-For header
    option forwardfor
# Define the back-end servers, which can handle up to 512 concurrent connections each
    server websvr1 192.168.1.71:80 weight 1 maxconn 512 check
    server websvr2 192.168.1.72:80 weight 1 maxconn 512 check
```

It is also possible to configure HAProxy and Keepalived directly on the web servers as shown in Figure 17.6. As in the previous example, one HAProxy server (10.0.0.11) is configured as the Keepalived master server with the virtual IP address 10.0.0.10 and the other (10.0.0.12) is configured as a Keepalived backup server. The HAProxy service on the master listens on port 80 and forwards incoming requests to one of the `httpd` services, which listen on port 8080.

Figure 17.6 Example of a Combined HAProxy and Keepalived Configuration with Integrated Web Servers
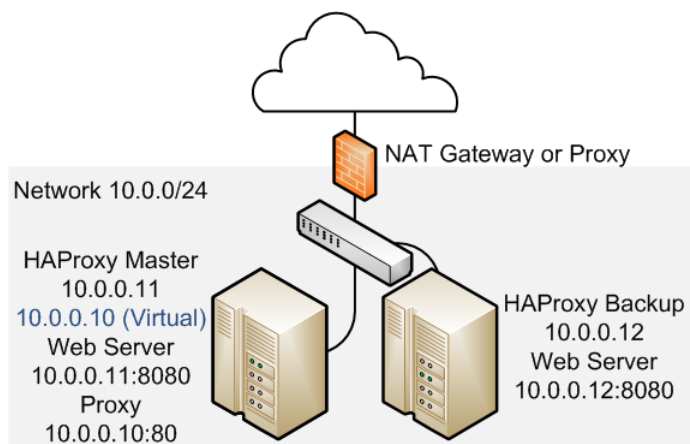
The HAProxy configuration is the same as the previous example except for the IP addresses and ports of the web servers.

```
...
    server websvr1 10.0.0.11:8080 weight 1 maxconn 512 check
    server websvr2 10.0.0.12:8080 weight 1 maxconn 512 check
```

The firewall on each server must be configured to accept incoming TCP requests on port 8080.

The Keepalived configuration for both example configurations is similar to that given in Section 17.6, "Configuring Simple Virtual IP Address Failover Using Keepalived".

The master server has the following Keepalived configuration:

```
global_defs {
   notification_email {
     root@mydomain.com
   }
   notification_email_from haproxy1@mydomain.com
   smtp_server localhost
   smtp_connect_timeout 30
}

vrrp_instance VRRP1 {
    state MASTER
#   Specify the network interface to which the virtual address is assigned
    interface enp0s8
#   The virtual router ID must be unique to each VRRP instance that you define
    virtual_router_id 41
#   Set the value of priority higher on the master server than on a backup server
```

```
    priority 200
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1066
    }
    virtual_ipaddress {
        10.0.0.10/24
    }
}
```

The configuration of the backup server is the same except for the values of `notification_email_from`, `state`, `priority`, and possibly `interface` if the system hardware configuration is different:

```
global_defs {
   notification_email {
     root@mydomain.com
   }
   notification_email_from haproxy2@mydomain.com
   smtp_server localhost
   smtp_connect_timeout 30
}

vrrp_instance VRRP1 {
    state BACKUP
#   Specify the network interface to which the virtual address is assigned
    interface enp0s8
    virtual_router_id 41
#   Set the value of priority lower on the backup server than on the master server
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1066
    }
    virtual_ipaddress {
        10.0.0.10/24
    }
}
```

In the event that the master server (`haproxy1`) fails, `keepalived` assigns the virtual IP address 10.0.0.10/24 to the `enp0s8` interface on the backup server (`haproxy2`), which becomes the master server.

See Section 17.2, "Installing and Configuring HAProxy" and Section 17.5, "Installing and Configuring Keepalived" for details of how to install and configure HAProxy and Keepalived.

17.11 About Keepalived Notification and Tracking Scripts

Notification scripts are executable programs that Keepalived invokes when a server changes state. You can implements notification scripts to perform actions such as reconfiguring a network interface or starting, reloading or stopping a service.

To invoke a notification script, include one the following lines inside a `vrrp_instance` or `vrrp_sync_group` section:

`notify program_path`   Invokes `program_path` with the following arguments:

  `$1` Set to `INSTANCE` or `GROUP`, depending on whether Keepalived invoked the program from `vrrp_instance` or `vrrp_sync_group`.

$2   Set to the name of the `vrrp_instance` or `vrrp_sync_group`.

$3   Set to the end state of the transition: `BACKUP`, `FAULT`, or `MASTER`.

`notify_backup` *`program_path`*, `notify_backup` `"`*`program_path arg`* `..."`	Invokes *`program_path`* when the end state of a transition is `BACKUP`. *`program_path`* is the full pathname of an executable script or binary. If a program has arguments, enclose both the program path and the arguments in quotes.
`notify_fault` *`program_path`*, `notify_fault` `"`*`program_path arg`* `..."`	Invokes *`program_path`* when the end state of a transition is `FAULT`.
`notify_master` *`program_path`*, `notify_master` `"`*`program_path arg`* `..."`	Invokes *`program_path`* when the end state of a transition is `MASTER`.

The following executable script could be used to handle the general-purpose version of `notify`:

```bash
#!/bin/bash

ENDSTATE=$3
NAME=$2
TYPE=$1

case $ENDSTATE in
    "BACKUP") # Perform action for transition to BACKUP state
              exit 0
              ;;
    "FAULT")  # Perform action for transition to FAULT state
              exit 0
              ;;
    "MASTER") # Perform action for transition to MASTER state
              exit 0
              ;;
    *)        echo "Unknown state ${ENDSTATE} for VRRP ${TYPE} ${NAME}"
              exit 1
              ;;
esac
```

Tracking scripts are programs that Keepalived runs at regular intervals, according to a `vrrp_script` definition:

```
vrrp_script script_name {
  script       "program_path arg ..."
  interval i   # Run script every i seconds
  fall f       # If script returns non-zero f times in succession, enter FAULT state
  rise r       # If script returns zero r times in succession, exit FAULT state
  timeout t    # Wait up to t seconds for script before assuming non-zero exit code
  weight w     # Reduce priority by w on fall
}
```

`program_path` is the full pathname of an executable script or binary.

You can use tracking scripts with a `vrrp_instance` section by specifying a `track_script` clause, for example:

```
vrrp_instance instance_name {
```

```
    state MASTER
    interface enp0s8
    virtual_router_id 21
    priority 200
    advert_int 1
    virtual_ipaddress {
       10.0.0.10/24
    }
    track_script {
       script_name
       ...
    }
}
```

If a configured script returns a non-zero exit code `f` times in succession, Keepalived changes the state of the VRRP instance or group to `FAULT`, removes the virtual IP address 10.0.0.10 from `enp0s8`, reduces the priority value by `w` and stops sending multicast VRRP packets. If the script subsequently returns a zero exit code `r` times in succession, the VRRP instance or group exits the `FAULT` state and transitions to the `MASTER` or `BACKUP` state depending on its new priority.

If you want a server to enter the `FAULT` state if one or more interfaces goes down, you can also use a `track_interface` clause, for example:

```
    track_interface {
       enp0s8
       enp0s9
    }
```

A possible application of tracking scripts is to deal with a potential split-brain condition in the case that some of the Keepalived servers lose communication. For example, a script could track the existence of other Keepalived servers or use shared storage or a backup communication channel to implement a voting mechanism. However, configuring Keepalived to avoid a split brain condition is complex and it is difficult to avoid corner cases where a scripted solution might not work.

For an alternative solution, see Section 17.12, "Making HAProxy Highly Available Using Oracle Clusterware".

17.12 Making HAProxy Highly Available Using Oracle Clusterware

When Keepalived is used with two or more servers, loss of network connectivity can result in a split-brain condition, where more than one server acts as the master, and which can result in data corruption. To avoid this scenario, Oracle recommends that you use HAProxy in conjunction with a *shoot the other node in the head* (STONITH) solution such as Oracle Clusterware to support virtual IP address failover in preference to Keepalived.

Oracle Clusterware is a portable clustering software solution that allow you to configure independent servers so that they cooperate as a single cluster. The individual servers within the cluster cooperate so that they appear to be a single server to external client applications.

The following example uses Oracle Clusterware with HAProxy for load balancing to HTTPD web server instances on each cluster node. In the event that the node running HAProxy and an HTTPD instance fails, the services and their virtual IP addresses fail over to the other cluster node.

Figure 17.7 shows two cluster nodes, which are connected to an externally facing network. The nodes are also linked by a private network that is used for the cluster heartbeat. The nodes have shared access to certified SAN or NAS storage that holds the voting disk and Oracle Cluster Registry (OCR) in addition to service configuration data and application data.

Figure 17.7 Example of an Oracle Clusterware Configuration with Two Nodes
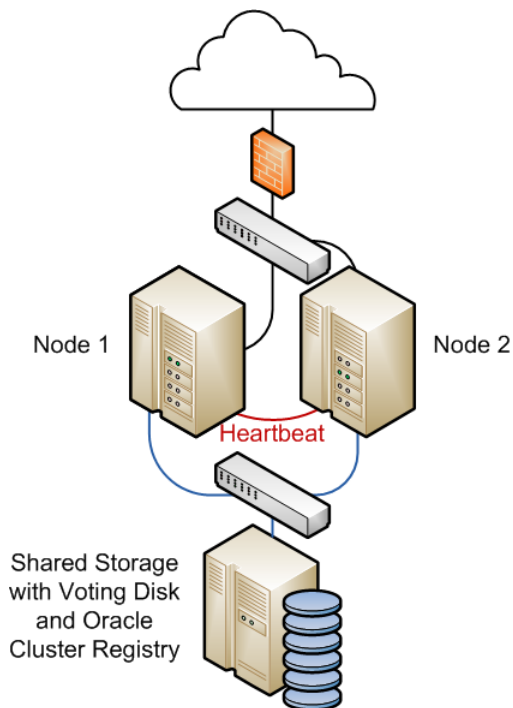
For a high-availability configuration, Oracle recommends that the network, heartbeat, and storage connections are multiply redundant and that at least three voting disks are configured.

The following steps outline how to configure such a cluster:

1. Install Oracle Clusterware on each system that will serve as a cluster node.

2. Install the `haproxy` and `httpd` packages on each node.

3. Use the `appvipcfg` command to create a virtual IP address for HAProxy and a separate virtual IP address for each HTTPD service instance. For example, if there are two HTTPD service instances, you would need to create three different virtual IP addresses.

4. Implement cluster scripts to start, stop, clean, and check the HAProxy and HTTPD services on each node. These scripts must return 0 for success and 1 for failure.

5. Use the shared storage to share the configuration files, HTML files, logs, and all directories and files that the HAProxy and HTTPD services on each node require to start.

    If you have an Oracle Linux Support subscription, you can use OCFS2 or ASM/ACFS with the shared storage as an alternative to NFS or other type of shared file system.

6. Configure each HTTPD service instance so that it binds to the correct virtual IP address. Each service instance must also have an independent set of configuration, log, and other required files, so that all of the service instances can coexist on the same server if one node fails.

7. Use the `crsctl` command to create a cluster resource for HAProxy and for each HTTPD service instance. If there are two or more HTTPD service instances, binding of these instances should initially be distributed amongst the cluster nodes. The HAProxy service can be started on either node initially.

You can use Oracle Clusterware as the basis of a more complex solution that protects a multi-tiered system consisting of front-end load balancers, web servers, database servers and other components.

For more information, see the Oracle Clusterware 11g Administration and Deployment Guide and the Oracle Clusterware 12c Administration and Deployment Guide.

Chapter 18 VNC Service Configuration

Table of Contents

This chapter describes how to enable a Virtual Network Computing (VNC) server to provide remote access to a graphical desktop.

18.1 About VNC

Virtual Network Computing (VNC) is a system for sharing a graphical desktop over a network. A VNC client (the "viewer") connects to, and can control, a desktop that is shared by a VNC server on a remote system. Because VNC is platform independent, you can use any operating system with a VNC client to connect to a VNC server. VNC makes remote administration using graphical tools possible.

By default, all communication between a VNC client and a VNC server is not secure. You can secure VNC communication by using an SSH tunnel. Using an SSH tunnel also reduces the number of firewall ports that need to be open. Oracle recommends that you use SSH tunnels.

18.2 Configuring a VNC Server

To configure a VNC server:

1. Install the `tigervnc-server` package:

   ```
   # yum install tigervnc-server
   ```

2. Create the VNC environment for the VNC users.

   Each VNC desktop on the system runs a VNC server as a particular user. This user must be able to log in to the system with a user name and either a password or an SSH key (if the VNC desktop is to be accessed through an SSH tunnel).

   Use the `vncpasswd` command to create a password for the VNC desktop. The password must be created by the user that runs the VNC server and not `root`, for example:

   ```
   # su - vncuser
   $ vncpasswd
   Password: password
   Verify: password
   ```

   The password must contain at least six characters. If the password is longer than eight characters, only the first eight characters are used for authentication. An obfuscated version of the password is stored in `$HOME/.vnc/passwd` unless the name of a file is specified with the `vncpasswd` command.

3. Create a service unit configuration file for each VNC desktop that is to be made available on the system.

   a. Copy the `vncserver@.service` template file, for example:

      ```
      # cp /lib/systemd/system/vncserver@.service \
      /etc/systemd/system/vncserver@\:display.service
      ```

where *display* is the unique display number of the VNC desktop starting from 1. Use a backslash character (`\`) to escape the colon (`:`) character.

Each VNC desktop is associated with a user account. For ease of administration if you have multiple VNC desktops, you can include the name of the VNC user in the name of the service unit configuration file, for example:

```
# cp /lib/systemd/system/vncserver@.service \
/etc/systemd/system/vncserver-vncuser@\:display.service
```

b. Edit the service unit configuration files.

Replace any instances of `<USER>` with the user name of the user that will run the VNC desktop, for example:

```
ExecStart=/sbin/runuser -l vncuser -c "/usr/bin/vncserver %i"
PIDFile=/home/vncuser/.vnc/%H%i.pid
```

Optionally, you can add command-line arguments for the VNC server. In the following example, the VNC server only accepts connections from `localhost`, which means the VNC desktop can only be accessed locally or through an SSH tunnel; and the size of the window has been changed from the default 1024x768 to 640x480 using the `geometry` flag:

```
ExecStart=/sbin/runuser -l vncuser -c "/usr/bin/vncserver %i -localhost -geometry 640x480"
PIDFile=/home/vncuser/.vnc/%H%i.pid
```

4. Start the VNC desktops.

a. Make `systemd` reload its configuration files:

```
# systemctl daemon-reload
```

b. For each VNC desktop, start the service, and configure the service to start following a system reboot. Remember that if you specified a username in the name of the service unit configuration file, you must specify this. Equally, you should use the same display number that you specified for the service unit configuration file name. For example:

```
# systemctl start vncserver-vncuser@\:display.service
# systemctl enable vncserver-vncuser@\:display.service
```

> **Note**
>
> If you make any changes to a service unit configuration file, you must reload the configuration file and restart the service.

5. Configure the firewall to allow access to the VNC desktops.

If users will access the VNC desktops through an SSH tunnel and the SSH service is enabled on the system, you do not need to open additional ports in the firewall. SSH is enabled by default. For information on enabling SSH, see Section 27.3, "Configuring an OpenSSH Server".

If users will access the VNC desktops directly, you must open the required port for each desktop. The required ports can be calculated by adding the VNC desktop service display number to 5900 (the default VNC server port). So if the display number is 1, the required port is 5901 and if the display number is 67, the required port is 5967.

To open ports 5900 to 5903, you can use the following commands:

```
# firewall-cmd --zone=zone --add-service=vnc-server
# firewall-cmd --zone=zone --add-service=vnc-server --permanent
```

To open additional ports, for example port 5967, use the following commands:

```
# firewall-cmd --zone=zone --add-port=5967/tcp
# firewall-cmd --zone=zone --add-port=5967/tcp --permanent
```

6. Configure the VNC desktops.

   By default, the VNC server runs the user's default desktop environment. This is controlled by the VNC user's `$HOME/.vnc/xstartup` file, which is created automatically when the VNC desktop service is started.

   If you did not install a desktop environment when you installed the system (for example because you selected Minimal Install as the base environment), you can install one with the following command:

   ```
   # yum groupinstall "server with gui"
   ```

   When the installation is complete, use the `systemctl get-default` command to check that the default system state is `multi-user.target` (multi-user command-line environment). Use the `systemctl set-default` command reset the default system state or to change it to the `graphical.target` (multi-user graphical environment) if you prefer.

   The `$HOME/.vnc/xstartup` file is a shell script that specifies the X applications to run when the VNC desktop is started. For example, to run a KDE Plasma Workspace, you could edit the file as follows:

   ```
   #!/bin/sh
   unset SESSION_MANAGER
   unset DBUS_SESSION_BUS_ADDRESS
   #exec /etc/X11/xinit/xinitrc
   startkde &
   ```

   If you make any changes to a user's `$HOME/.vnc/xstartup` file, you must restart the VNC desktop for the changes to take effect:

   ```
   # systemctl restart vncserver-vncuser@\:display.service
   ```

   See the `vncserver(1)`, `Xvnc(1)`, and `vncpasswd(1)` manual pages for more information.

18.3 Connecting to VNC Desktop

You can connect to a VNC desktop on an Oracle Linux 7 system using any VNC client. The following example instructions are for the TigerVNC client. Adapt the instructions for your client.

On Linux platforms:

1. Install the TigerVNC client (`vncviewer`).

   ```
   # yum install tigervnc
   ```

2. Start the TigerVNC client and connect to a desktop.

   To connect directly to a VNC desktop, you can start the TigerVNC client and enter `host:display` to specify the host name or IP address of the VNC server and the display number of the VNC desktop to connect to. Alternatively, you can specify the VNC desktop as an argument for the `vncviewer` command. For example:

   ```
   $ vncviewer myhost.example.com:1
   ```

To connect to a VNC desktop through an SSH tunnel, use the `-via` option for the `vncviewer` command to specify the user name and host for the SSH connection, and use `localhost:`*`display`* to specify the VNC desktop. For example:

```
$ vncviewer -via vncuser@myhost.example.com localhost:67
```

See the `vncviewer(1)` manual page for more information.

On Microsoft Windows platforms:

1. Download and install the TigerVNC client (`vncviewer.exe`) from http://tigervnc.org.

2. Start the TigerVNC client and connect to a desktop.

   To connect directly to a VNC desktop, start the TigerVNC client and enter *`host:display`* to specify the host name or IP address of the VNC server and the display number of the VNC desktop to connect to.

   To connect to a VNC desktop through an SSH tunnel, requires the use of an SSH client program such as PuTTY. For example:

   a. Start PuTTY and create a new SSH connection to the VNC server.

      In the PuTTY Configuration window, navigate to **Session**, and enter the host name or IP address and port.

   b. Enable X11 forwarding.

      In the PuTTY Configuration window, navigate to **Connection**, **SSH**, and **X11**, and then select **Enable X11 forwarding**.

   c. Create the SSH tunnel.

      In the PuTTY Configuration window, navigate to **Connection**, **SSH**, and **Tunnels**. In the **Source port** box enter the port number on the client that is to be forwarded, for example `5900`. In the **Destination** box enter *`host:display`* to specify the host name or IP address of the VNC server and the display number of the VNC desktop to connect to. Then click **Add**.

   d. Save the configuration.

      In the PuTTY Configuration window, navigate to **Session**, enter a name for the session in the **Saved sessions** box and click **Save**.

   e. Select the saved session, click **Load** and then click *Open*, and establish an SSH connection to the VNC server host.

   f. Start the TigerVNC client, and connect to `localhost:`*`display`*, where *`display`* is the source port number configured in the SSH tunnel. You might have to configure the firewall on the client to permit the connection.

Part III Storage and File Systems

This section contains the following chapters:

- Chapter 19, *Storage Management* describes how to configure and manage disk partitions, swap space, logical volumes, software RAID, block device encryption, iSCSI storage, and multipathing.

- Chapter 20, *File System Administration* describes how to create, mount, check, and repair file systems, how to configure Access Control Lists, how to configure and manage disk quotas.

- Chapter 21, *Local File System Administration* describes administration tasks for the btrfs, ext3, ext4, OCFS2, and XFS local file systems.

- Chapter 22, *Shared File System Administration* describes administration tasks for the NFS and Samba shared file systems, including how to configure NFS and Samba servers.

- Chapter 23, *Oracle Cluster File System Version 2* describes how to configure and use the Oracle Cluster File System Version 2 (OCFS2) file system.

Table of Contents

Chapter 19 Storage Management

Table of Contents

This chapter describes how to configure and manage disk partitions, swap space, logical volumes, software RAID, block device encryption, iSCSI storage, and multipathing.

19.1 About Disk Partitions

Partitioning a disk drive divides it into one or more reserved areas (*partitions*) and stores information about these partitions in the partition table on the disk The operating system treats each partition as a separate disk that can contain a file system.

Oracle Linux requires one partition for the root file system. It is usual to use two other partitions for swap space and the boot file system. On x86 and x86_64 systems, the system BIOS can usually access only the first 1024 cylinders of the disk at boot time. Configuring a separate boot partition in this region on the disk allows the GRUB bootloader to access the kernel image and other files that are required to boot the system.

You can create additional partitions to simplify backups, to enhance system security, and to meet other needs, such as setting up development sandboxes and test areas. Data that frequently changes, such as user home directories, databases, and log file directories, is typically assigned to separate partitions to facilitate backups.

The partitioning scheme for hard disks with a master boot record (MBR) allows you to create up to four *primary partitions*. If you need more than four partitions, you can divide one of the primary partitions into

up to 11 *logical partitions*. The primary partition that contains the logical partitions is known as an *extended partition*. The MBR scheme supports disks up to 2 TB in size.

On hard disks with a GUID Partition Table (GPT), you can configure up to 128 partitions and there is no concept of extended or logical partitions. You should configure a GPT if the disk is larger than 2 TB.

You can create and manage MBRs by using the `fdisk` command. If you want to create a GPT, use `parted` instead.

> **Note**
>
> When partitioning a block storage device, align primary and logical partitions on one-megabyte (1048576 bytes) boundaries. If partitions, file system blocks, or RAID stripes are incorrectly aligned and overlap the boundaries of the underlying storage's sectors or pages, the device controller has to modify twice as many sectors or pages than if correct alignment is used. This recommendation applies to most block storage devices, including hard disk drives (*spinning rust*), solid state drives (SSDs), LUNs on storage arrays, and host RAID adapters.

19.1.1 Managing Partition Tables Using fdisk

> **Caution**
>
> If any partition on the disk to be configured using `fdisk` is currently mounted, unmount it before running `fdisk` on the disk. Similarly, if any partition is being used as swap space, use the `swapoff` command to disable the partition.
>
> Before running `fdisk` on a disk that contains data, first back up the data on to another disk or medium.
>
> You cannot use `fdisk` to manage a GPT hard disk.

You can use the `fdisk` utility to create a partition table, view an existing partition table, add partitions, and delete partitions. Alternatively, you can also use the `cfdisk` utility, which is a text-based, graphical version of `fdisk`.

You can use `fdisk` interactively or you can use command-line options and arguments to specify partitions. When you run `fdisk` interactively, you specify only the name of the disk device as an argument, for example:

```
# fdisk /dev/sda

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
         switch off the mode (command 'c') and change display units to
         sectors (command 'u').

Command (m for help):
```

If you disable DOS-compatibility mode, `fdisk` aligns partitions on one-megabyte boundaries. It is recommended that you turn off DOS-compatibility mode and use display units of 512-byte sectors by specifying the `-c` and `-u` options or by entering the `c` and `u` commands.

Enter `c` to switch off DOS-compatibility mode, `u` to use sectors, and `p` to display the partition table:

```
Command (m for help): c
DOS Compatibility flag is not set

Command (m for help): u
Changing display/entry units to sectors
```

```
Command (m for help): p

Disk /dev/sda: 42.9 GB, 42949672960 bytes
255 heads, 63 sectors/track, 5221 cylinders, total 83886080 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x0002a95d

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048     1026047      512000   83  Linux
/dev/sda2         1026048    83886079    41430016   8e  Linux LVM
```

The example output shows that `/dev/sda` is a 42.9 GB disk. As modern hard disks support logical block addressing (LBA), any information about the numbers of heads and sectors per track is irrelevant and probably fictitious. The start and end offsets of each partition from the beginning of the disk are shown in units of sectors. The partition table is displayed after the device summary, and shows:

`Device`	The device that corresponds to the partition.
`Boot`	Specifies `*` if the partition contains the files that the GRUB bootloader needs to boot the system. Only one partition can be bootable.
`Start` and `End`	The start and end offsets in sectors. All partitions are aligned on one-megabyte boundaries.
`Blocks`	The size of the partition in one-kilobyte blocks.
`Id` and `System`	The partition type. The following partition types are typically used with Oracle Linux:

`5 Extended`	An extended partition that can contain up to four logical partitions.
`82 Linux swap`	Swap space partition.
`83 Linux`	Linux partition for a file system that is not managed by LVM. This is the default partition type.
`8e Linux LVM`	Linux partition that is managed by LVM.

The `n` command creates a new partition. For example, to create partition table entries for two Linux partitions on `/dev/sdc`, one of which is 5 GB in size and the other occupies the remainder of the disk:

```
# fdisk -cu /dev/sdc
...
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First sector (2048-25165823, default 2048):  2048
Last sector, +sectors or +size{K,M,G} (2048-25165823, default 25165823): +5G

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
```

```
p
Partition number (1-4): 2
First sector (10487808-25165823, default 10487808): <Enter>
Using default value 10487808
Last sector, +sectors or +size{K,M,G} (10487808-25165823, default 25165823): <Enter>
Using default value 25165823

Command (m for help): p

Disk /dev/sdc: 12.9 GB, 12884901888 bytes
255 heads, 63 sectors/track, 1566 cylinders, total 25165824 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xe6d3c9f6

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1            2048    10487807     5242880   83  Linux
/dev/sdc2        10487808    25165823     7339008   83  Linux
```

The `t` command allows you to change the type of a partition. For example, to change the partition type of partition 2 to `Linux LVM`:

```
Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): 8e

Command (m for help): p
...
   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1            2048    10487807     5242880   83  Linux
/dev/sdc2        10487808    25165823     7339008   8e  Linux LVM
```

After creating the new partition table, use the `w` command to write the table to the disk and exit `fdisk`.

```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

If you enter `q` instead, `fdisk` exits without committing the changes to disk.

For more information, see the `cfdisk(8)` and `fdisk(8)` manual pages.

19.1.2 Managing Partition Tables Using parted

⚠️

Caution

If any partition on the disk to be configured using `parted` is currently mounted, unmount it before running `parted` on the disk. Similarly, if any partition is being used as swap space, use the `swapoff` command to disable the partition.

Before running `parted` on a disk that contains data, first back up the data on to another disk or medium.

You can use the `parted` utility to label a disk, create a partition table, view an existing partition table, add partitions, change the size of partitions, and delete partitions. `parted` is more advanced than `fdisk` as it supports more disk label types, including GPT disks, and it implements a larger set of commands.

You can use `parted` interactively or you can specify commands as arguments. When you run `parted` interactively, you specify only the name of the disk device as an argument, for example:

```
# parted /dev/sda
GNU Parted 2.1
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted)
```

The `print` command displays the partition table:

```
(parted) print
Model: ATA VBOX HARDDISK (scsi)
Disk /dev/sda: 42.9GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos

Number  Start   End     Size    Type     File system  Flags
 1      1049kB  525MB   524MB   primary  ext4         boot
 2      525MB   42.9GB  42.4GB  primary               lvm
```

The `mklabel` command creates a new partition table:

```
# parted /dev/sdd
GNU Parted 2.1
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mklabel
New disk label type? gpt
Warning: The existing disk label on /dev/sdd will be destroyed
and all data on this disk will be lost. Do you want to continue?
Yes/No? y
```

Typically, you would set the disk label type to `gpt` or `msdos` for an Oracle Linux system, depending on whether the disk device supports GPT. You are prompted to confirm that you want to overwrite the existing disk label.

The `mkpart` command creates a new partition:

```
(parted) mkpart
Partition name? []? <Enter>
File system type? [ext2]? ext4
Start? 1
End? 5GB
```

For disks with an `msdos` label, you are also prompted to enter the partition type, which can be `primary`, `extended`, or `logical`. The file system type is typically set to one of `fat16`, `fat32`, `ext4`, or `linux-swap` for an Oracle Linux system. If you are going to create an `btrfs`, `ext*`, `ocfs2`, or `xfs` file system on the partition, specify `ext4`. Unless you specify units such as GB for gigabytes, the start and end offsets of a partition are assumed to be in megabytes. To specify the end of the disk for `End`, enter a value of `-0`.

To display the new partition, use the `print` command:

```
(parted) print
Number  Start   End     Size    File system  Name  Flags
1       1049kB  5000MB  4999MB  ext4
```

To exit parted, enter `quit`.

> **Note**
>
> `parted` commands such as `mklabel` and `mkpart` commit the changes to disk immediately. Unlike `fdisk`, you do not have the option of quitting without saving your changes.

For more information, see the `parted(8)` manual page or enter `info parted` to view the online user manual.

19.1.3 Mapping Partition Tables to Devices

You can use the `kpartx` utility to map the partitions of any block device or file that contains a partition table and partition images. `kpartx` reads the partition table and creates device files for the partitions in `/dev/mapper`. Each device file represents a disk volume or a disk partition on a device or within an image file.

The `-l` option lists any partitions that it finds, for example in an installation image file:

```
# kpartx -l system.img
loop0p1 : 0 204800 /dev/loop0 2048
loop0p2 : 0 12288000 /dev/loop0 206848
loop0p3 : 0 4096000 /dev/loop0 212494848
loop0p4 : 0 2 /dev/loop0 16590848
```

This output shows that the drive image contains four partitions, and the first column are the names of the device files that can be created in `/dev/mapper`.

The `-a` option creates the device mappings:

```
# kpartx -a system.img
# ls /dev/mapper
control  loop0p1  loop0p2  loop0p3  loop0p4
```

If a partition contains a file system, you can mount it and view the files that it contains, for example:

```
# mkdir /mnt/sysimage
# mount /dev/mapper/loop0p1 /mnt/sysimage
# ls /mnt/sysimage
config-2.6.32-220.el6.x86_64
config-2.6.32-300.3.1.el6uek.x86_64
efi
grub
initramfs-2.6.32-220.el6.x86_64.img
initramfs-2.6.32-300.3.1.el6uek.x86_64.img
...
# umount /mnt/sysimage
```

The `-d` option removes the device mappings:

```
# kpartx -d system.img
# ls /dev/mapper
control
```

For more information, see the `kpartx(8)` manual page.

19.2 About Swap Space

Oracle Linux uses swap space when your system does not have enough physical memory to store the text (code) and data pages that the processes are currently using. When your system needs more memory, it writes inactive pages to swap space on disk, freeing up physical memory. However, writing to swap space has a negative impact on system performance, so increasing swap space is not an effective solution to shortage of memory. Swap space is located on disk drives, which have much slower access times than physical memory. If your system often resorts to swapping, you should add more physical memory, not more swap space.

You can configure swap space on a swap file in a file system or on a separate swap partition. A dedicated swap partition is faster, but changing the size of a swap file is easier. Configure a swap partition if you

know how much swap space your system requires. Otherwise, start with a swap file and create a swap partition when you know what your system requires.

19.2.1 Viewing Swap Space Usage

To view a system's usage of swap space, examine the contents of `/proc/swaps`:

```
# cat /proc/swaps
Filename                Type        Size      Used    Priority
/dev/sda2               partition   4128760   388     -1
/swapfile               file        999992    0       -2
```

In this example, the system is using both a 4-gigabyte swap partition on `/dev/sda2` and a one-gigabyte swap file, `/swapfile`. The `Priority` column shows that the system preferentially swaps to the swap partition rather than to the swap file.

You can also view `/proc/meminfo` or use utilities such as `free`, `top`, and `vmstat` to view swap space usage, for example:

```
# grep Swap /proc/meminfo
SwapCached:           248 kB
SwapTotal:        5128752 kB
SwapFree:         5128364 kB
# free | grep Swap
Swap:      5128752        388    5128364
```

19.2.2 Creating and Using a Swap File

> **Note**
>
> Configuring a swap file on a btrfs file system is not supported.

To create and use a swap file:

1. Use the `dd` command to create a file of the required size (for example, one million one-kilobyte blocks):

   ```
   # dd if=/dev/zero of=/swapfile bs=1024 count=1000000
   ```

2. Initialize the file as a swap file:

   ```
   # mkswap /swapfile
   ```

3. Enable swapping to the swap file:

   ```
   # swapon /swapfile
   ```

4. Add an entry to `/etc/fstab` for the swap file so that the system uses it following the next reboot:

   ```
   /swapfile       swap      swap       defaults      0 0
   ```

19.2.3 Creating and Using a Swap Partition

To create and use a swap partition:

1. Use `fdisk` to create a disk partition of type `82` (`Linux swap`) or `parted` to create a disk partition of type `linux-swap` of the size that you require.

2. Initialize the partition (for example, `/dev/sda2`) as a swap partition:

   ```
   # mkswap /dev/sda2
   ```

3. Enable swapping to the swap partition:

```
# swapon /swapfile
```

4. Add an entry to `/etc/fstab` for the swap partition so that the system uses it following the next reboot:

```
/dev/sda2        swap       swap       defaults        0 0
```

19.2.4 Removing a Swap File or Swap Partition

To remove a swap file or swap partition from use:

1. Disable swapping to the swap file or swap partition, for example:

```
# swapoff /swapfile
```

2. Remove the entry for the swap file or swap partition from `/etc/fstab`.

3. Optionally, remove the swap file or swap partition if you do not want to use it in future.

19.3 About Logical Volume Manager

You can use Logical Volume Manager (LVM) to manage multiple physical volumes and configure mirroring and striping of logical volumes to provide data redundancy and increase I/O performance. In LVM, you first create volume groups from physical volumes, which are storage devices such as disk array LUNs, software or hardware RAID devices, hard drives, and disk partitions. You can then create logical volumes in a volume group. A logical volume functions as a partition that in its implementation might be spread over multiple physical disks.

You can create file systems on logical volumes and mount the logical volume devices in the same way as you would a physical device. If a file system on a logical volume becomes full with data, you can increase the capacity of the volume by using free space in the volume group so that you can then grow the file system (provided that the file system has that capability). If necessary, you can add physical storage devices to a volume group to increase its capacity.

LVM is non-disruptive and transparent to users. You can increase the size of logical volumes and change their layout dynamically without needing to schedule system down time to reconfigure physical storage.

LVM uses the device mapper (DM) that provides an abstraction layer that allows the creation of logical devices above physical devices and provides the foundation for software RAID, encryption, and other storage features.

19.3.1 Initializing and Managing Physical Volumes

Before you can create a volume group, you must initialize the physical devices that you want to use as physical volumes with LVM.

> ⚠️ **Caution**
>
> If the devices contain any existing data, back up the data.

To set up a physical device as a physical volume, use the `pvcreate` command:

```
# pvcreate [options] device ...
```

For example, set up `/dev/sdb`, `/dev/sdc`, `/dev/sdd`, and `/dev/sde` as physical volumes:

```
# pvcreate -v /dev/sd[bcde]
Set up physical volume for "/dev/sdb" with 6313482 available
sectors
Zeroing start of device /dev/sdb
Physical volume "/dev/sdb" successfully created
...
```

To display information about physical volumes, you can use the `pvdisplay`, `pvs`, and `pvscan` commands.

To remove a physical volume from the control of LVM, use the `pvremove` command:

```
# pvremove device
```

Other commands that are available for managing physical volumes include `pvchange`, `pvck`, `pvmove`, and `pvresize`.

For more information, see the `lvm(8)`, `pvcreate(8)`, and other LVM manual pages.

19.3.2 Creating and Managing Volume Groups

Having initialized the physical volumes, you can add them to a new or existing volume group.

To create a volume group, use the `vgcreate` command:

```
# vgcreate [options] volume_group physical_volume ...
```

For example, create the volume group `myvg` from the physical volumes `/dev/sdb`, `/dev/sdc`, `/dev/sdd`, and `/dev/sde`:

```
# vgcreate -v myvg /dev/sd[bcde]
Wiping cache of LVM-capable devices
Adding physical volume '/dev/sdb' to volume group 'myvg'
Adding physical volume '/dev/sdc' to volume group 'myvg'
Adding physical volume '/dev/sdd' to volume group 'myvg'
Adding physical volume '/dev/sde' to volume group 'myvg'
Archiving volume group "myvg" metadata (seqno 0).
Creating volume group backup "/etc/lvm/backup/myvg" (seqno 1).
Volume group "myvg" successfully created
```

LVM divides the storage space within a volume group into physical *extents*, which are the smallest unit that LVM uses when allocating storage to logical volumes. The default size of an extent is 4 MB.

The *allocation policy* for the volume group and logical volume determines how LVM allocates extents from a volume group. The default allocation policy for a volume group is `normal`, which applies rules such as not placing parallel stripes on the same physical volume. The default allocation policy for a logical volume is `inherit`, which means that the logical volume uses the same policy as for the volume group. You can change the default allocation policies by using the `lvchange` or `vgchange` commands, or you can override the allocation policy when you create a volume group or logical volume. Other allocation policies include `anywhere`, `contiguous` and `cling`.

To add physical volumes to a volume group, use the `vgextend` command:

```
# vgextend [options] volume_group physical_volume ...
```

To remove physical volumes from a volume group, use the `vgreduce` command:

```
# vgreduce [options] volume_group physical_volume ...
```

To display information about volume groups, you can use the `vgdisplay`, `vgs`, and `vgscan` commands.

To remove a volume group from LVM, use the `vgremove` command:

```
# vgremove volume_group
```

Other commands that are available for managing volume groups include `vgchange`, `vgck`, `vgexport`, `vgimport`, `vgmerge`, `vgrename`, and `vgsplit`.

For more information, see the `lvm(8)`, `vgcreate(8)`, and other LVM manual pages.

19.3.3 Creating and Managing Logical Volumes

Having create a volume group of physical volumes, you can create logical volumes from the storage space that is available in the volume group.

To create a logical volume, use the `lvcreate` command:

```
# lvcreate [options] --size size --name logical_volume volume_group
```

For example, create the logical volume `mylv` of size 2 GB in the volume group `myvg`:

```
# lvcreate -v --size 2g --name mylv myvg
Setting logging type to disk
Finding volume group "myvg"
Archiving volume group "myvg" metadata (seqno 1).
Creating logical volume mylv
Create volume group backup "/etc/lvm/backup/myvg" (seqno 2).
...
```

`lvcreate` uses the device mapper to create a block device file entry under `/dev` for each logical volume and uses udev to set up symbolic links to this device file from `/dev/mapper` and `/dev/volume_group`. For example, the device that corresponds to the logical volume `mylv` in the volume group `myvg` might be `/dev/dm-3`, which is symbolically linked by `/dev/mapper/myvolg-myvol` and `/dev/myvolg/myvol`.

> **Note**
>
> Always use the devices in `/dev/mapper` or `/dev/volume_group`. These names are persistent and are created automatically by the device mapper early in the boot process. The `/dev/dm-*` devices are not guaranteed to be persistent across reboots.

Having created a logical volume, you can configure and use it in the same way as you would a physical storage device. For example, you can configure a logical volume as a file system, swap partition, Automatic Storage Management (ASM) disk, or raw device.

To display information about logical volumes, you can use the `lvdisplay`, `lvs`, and `lvscan` commands.

To remove a logical volume from a volume group, use the `lvremove` command:

```
# lvremove volume_group/logical_volume
```

> **Note**
>
> You must specify both the name of the volume group and the logical volume.

Other commands that are available for managing logical volumes include `lvchange`, `lvconvert`, `lvmdiskscan`, `lvmsadc`, `lvmsar`, `lvrename`, and `lvresize`.

For more information, see the `lvm(8)`, `lvcreate(8)`, and other LVM manual pages.

19.3.4 Creating Logical Volume Snapshots

You can also use `lvcreate` with the `--snapshot` option to create a snapshot of an existing logical volume such as `mylv` in the volume group `myvg`, for example:

```
# lvcreate --size 500m --snapshot --name mylv-snapshot myvg/mylv
  Logical volume "mylv-snapshot" created
```

You can mount and modify the contents of the snapshot independently of the original volume or preserve it as a record of the state of the original volume at the time that you took the snapshot. The snapshot usually takes up less space than the original volume, depending on how much the contents of the volumes diverge over time. In the example, we assume that the snapshot only requires one quarter of the space of the original volume. You can use the value shown by the `Snap%` column in the output from the `lvs` command to see how much data is allocated to the snapshot. If the value of `Snap%` approaches 100%, indicating that a snapshot is running out of storage, use `lvresize` to grow it. Alternatively, you can reduce a snapshot's size to save storage space. To merge a snapshot with its original volume, use the `lvconvert` command, specifying the `--merge` option.

To remove a logical volume snapshot from a volume group, use the `lvremove` command as you would for a logical volume:

```
# lvremove volume_group/logical_volume_snapshot
```

For more information, see the `lvcreate(8)` and `lvremove (8)` manual pages.

19.3.5 Creating and Managing Thinly-Provisioned Logical Volumes

Thinly-provisioned logical volumes have virtual sizes that are typically greater than the physical storage on which you create them. You create thinly-provisioned logical volumes from storage that you have assigned to a special type of logical volume termed a *thin pool*. LVM assigns storage on demand from a thin pool to a thinly-provisioned logical volume as required by the applications that access the volume. You need to use the `lvs` command to monitor the usage of the thin pool so that you can increase its size if its available storage is in danger of being exhausted.

To create a thin pool, use the `lvcreate` command with the `--thin` option:

```
# lvcreate --size size --thin volume_group/thin_pool_name
```

For example, create the thin pool `mytp` of size 1 GB in the volume group `myvg`:

```
# lvcreate --size 1g --thin myvg/mytp
Logical volume "mytp" created
```

You can then use `lvcreate` with the `--thin` option to create a thinly-provisioned logical volume with a size specified by the `--virtualsize` option, for example:

```
# lvcreate --virtualsize size --thin volume_group/thin_pool_name \
  --name logical_volume
```

For example, create the thinly-provisioned logical volume `mytv` with a virtual size of 2 GB using the thin pool `mytp`, whose size is currently less than the size of the volume:

```
# lvcreate --virtualsize 2g --thin myvg/mytp --name mytv
Logical volume "mytv" created
```

If you create a thin snapshot of a thinly-provisioned logical volume, do not specify the size of the snapshot, for example:

```
# lvcreate --snapshot --name mytv-snapshot myvg/mytv
```

```
Logical volume "mytv-snapshot" created
```

If you were to specify a size for the thin snapshot, its storage would not be provisioned from the thin pool.

If there is sufficient space in the volume group, you can use the `lvresize` command to increase the size of a thin pool, for example:

```
# lvresize -L+1G myvg/mytp
  Extending logical volume mytp to 2 GiB
  Logical volume mytp successfully resized
```

For details of how to use the `snapper` command to create and manage thin snapshots, see Section 19.3.6, "Using snapper with Thinly-Provisioned Logical Volumes".

For more information, see the `lvcreate(8)` and `lvresize(8)` manual pages.

19.3.6 Using snapper with Thinly-Provisioned Logical Volumes

You can use the `snapper` utility to create and manage thin snapshots of thinly-provisioned logical volumes.

To set up the snapper configuration for an existing mounted volume:

```
# snapper -c config_name create-config -f "lvm(fs_type)" fs_name
```

Here `config_name` is the name of the configuration, `fs_type` is the file system type (`ext4` or `xfs`), and `fs_name` is the path of the file system. The command adds an entry for `config_name` to `/etc/sysconfig/snapper`, creates the configuration file `/etc/snapper/configs/config_name`, and sets up a `.snapshots` subdirectory for the snapshots.

By default, `snapper` sets up a `cron.hourly` job to create snapshots in the `.snapshot` subdirectory of the volume and a `cron.daily` job to clean up old snapshots. You can edit the configuration file to disable or change this behavior. For more information, see the `snapper-configs(5)` manual page.

There are three types of snapshot that you can create using `snapper`:

`post`	You use a *post snapshot* to record the state of a volume after a modification. A post snapshot should always be paired with a *pre snapshot* that you take immediately before you make the modification.
`pre`	You use a *pre snapshot* to record the state of a volume before a modification. A pre snapshot should always be paired with a *post snapshot* that you take immediately after you have completed the modification.
`single`	You can use a single snapshot to record the state of a volume but it does not have any association with other snapshots of the volume.

For example, the following commands create pre and post snapshots of a volume:

```
# snapper -c config_name create -t pre -p
N
... Modify the volume's contents ...
# snapper -c config_name create -t post --pre-num N -p
N'
```

The `-p` option causes `snapper` to display the number of the snapshot so that you can reference it when you create the post snapshot or when you compare the contents of the pre and post snapshots.

To display the files and directories that have been added, removed, or modified between the pre and post snapshots, use the `status` subcommand:

```
# snapper -c config_name status N..N'
```

To display the differences between the contents of the files in the pre and post snapshots, use the `diff` subcommand:

```
# snapper -c config_name diff N..N'
```

To list the snapshots that exist for a volume:

```
# snapper -c config_name list
```

To delete a snapshot, specify its number to the `delete` subcommand:

```
# snapper -c config_name delete N''
```

To undo the changes in the volume from post snapshot N' to pre snapshot N:

```
# snapper -c config_name undochange N..N'
```

For more information, see the `snapper(8)` manual page.

19.4 About Software RAID

The Redundant Array of Independent Disks (RAID) feature allows you to spread data across the drives to increase capacity, implement data redundancy, and increase performance. RAID is usually implemented either in hardware on intelligent disk storage that exports the RAID volumes as LUNs, or in software by the operating system. Oracle Linux kernel uses the multidisk (MD) driver to support software RAID by creating virtual devices from two or more physical storage devices. You can use MD to organize disk drives into RAID devices and implement different RAID levels.

The following software RAID levels are commonly used with Oracle Linux:

Linear RAID (spanning)	Combines drives as a larger virtual drive. There is no data redundancy or performance benefit. Resilience decreases because the failure of a single drive renders the array unusable.
RAID-0 (striping)	Increases performance but does not provide data redundancy. Data is broken down into units (stripes) and written to all the drives in the array. Resilience decreases because the failure of a single drive renders the array unusable.
RAID-1 (mirroring)	Provides data redundancy and resilience by writing identical data to each drive in the array. If one drive fails, a mirror can satisfy I/O requests. Mirroring is an expensive solution because the same information is written to all of the disks in the array.
RAID-5 (striping with distributed parity)	Increases read performance by using striping and provides data redundancy. The parity is distributed across all the drives in an array but it does not take up as much space as a complete mirror. Write performance is reduced to some extent from RAID-0 by having to calculate parity information and write this information in addition to the data. If one disk in the array fails, the parity information is used to reconstruct data to satisfy I/O requests. In this mode, read performance and resilience are degraded until you replace the failed drive and it is

repopulated with data and parity information. RAID-5 is intermediate in expense between RAID-0 and RAID-1.

RAID-6 (striping with double distributed parity)	A more resilient variant of RAID-5 that can recover from the loss of two drives in an array. RAID-6 is used when data redundancy and resilience are important, but performance is not. RAID-6 is intermediate in expense between RAID-5 and RAID-1.
RAID 0+1 (mirroring of striped disks)	Combines RAID-0 and RAID-1 by mirroring a striped array to provide both increased performance and data redundancy. Failure of a single disk causes one of the mirrors to be unusable until you replace the disk and repopulate it with data. Resilience is degraded while only a single mirror remains available. RAID 0+1 is usually as expensive as or slightly more expensive than RAID-1.
RAID 1+0 (striping of mirrored disks or RAID-10)	Combines RAID-0 and RAID-1 by striping a mirrored array to provide both increased performance and data redundancy. Failure of a single disk causes part of one mirror to be unusable until you replace the disk and repopulate it with data. Resilience is degraded while only a single mirror retains a complete copy of the data. RAID 1+0 is usually as expensive as or slightly more expensive than RAID-1.

19.4.1 Creating Software RAID Devices

To create a software RAID device:

1. Use the mdadm command to create the MD RAID device:

   ```
   # mdadm --create md_device --level=RAID_level [options] --raid-devices=N device ...
   ```

   For example, to create a RAID-1 device /dev/md0 from /dev/sdf and /dev/sdg:

   ```
   # mdadm --create /dev/md0 --level=1 -raid-devices=2 /dev/sd[fg]
   ```

   Create a RAID-5 device /dev/md1 from /dev/sdb, /dev/sdc, and dev/sdd:

   ```
   # mdadm --create /dev/md1 --level=5 -raid-devices=3 /dev/sd[bcd]
   ```

   If you want to include spare devices that are available for expansion, reconfiguration, or replacing failed drives, use the --spare-devices option to specify their number, for example:

   ```
   # mdadm --create /dev/md1 --level=5 -raid-devices=3 --spare-devices=1 /dev/sd[bcde]
   ```

   Note

   The number of RAID and spare devices must equal the number of devices that you specify.

2. Add the RAID configuration to /etc/mdadm.conf:

   ```
   # mdadm --examine --scan >> /etc/mdadm.conf
   ```

   Note

   This step is optional. It helps mdadm to assemble the arrays at boot time.

   For example, the following entries in /etc/mdadm.conf define the devices and arrays that correspond to /dev/md0 and /dev/md1:

```
DEVICE /dev/sd[c-g]
ARRAY /dev/md0 devices=/dev/sdf,/dev/sdg
ARRAY /dev/md1 spares=1 devices=/dev/sdb,/dev/sdc,/dev/sdd,/dev/sde
```

For more examples, see the sample configuration file `/usr/share/doc/mdadm-3.2.1/mdadm.conf-example`.

Having created an MD RAID device, you can configure and use it in the same way as you would a physical storage device. For example, you can configure it as an LVM physical volume, file system, swap partition, Automatic Storage Management (ASM) disk, or raw device.

You can view `/proc/mdstat` to check the status of the MD RAID devices, for example:

```
# cat /proc/mdstat
Personalities : [raid1]
mdo : active raid1 sdg[1] sdf[0]
```

To display summary and detailed information about MD RAID devices, you can use the `--query` and `--detail` options with `mdadm`.

For more information, see the `md(4)`, `mdadm(8)`, and `mdadm.conf(5)` manual pages.

19.5 Creating Encrypted Block Devices

The device mapper supports the creation of encrypted block devices using the `dm-crypt` device driver. You can access data on encrypted devices at boot time only if you enter the correct password. As the underlying block device is encrypted and not the file system, you can use `dm-crypt` to encrypt disk partitions, RAID volumes, and LVM physical volumes, regardless of their contents.

When you install Oracle Linux, you have the option of configure encryption on system volumes other than the partition from which the system boots. If you want to protect the bootable partition, consider using any password protection mechanism that is built into the BIOS or setting up a GRUB password.

You use the `cryptsetup` utility to set up Linux Unified Key Setup (LUKS) encryption on the device and to manage authentication.

To set up the mapped device for an encrypted volume:

1. Initialize a LUKS partition on the device and set up the initial key, for example:

```
# cryptsetup luksFormat /dev/sdd
WARNING!
========
This will overwrite data on /dev/sdd irrevocably.
Are you sure? (Type uppercase yes): YES
Enter LUKS passphrase: passphrase
Verify passphrase: passphrase
```

2. Open the device and create the device mapping:

```
# cryptsetup luksOpen /dev/sdd cryptfs
Enter passphrase for /dev/sdd: passphrase
```

In this example, the encrypted volume is accessible as `/dev/mapper/cryptsfs`.

3. Create an entry for the encrypted volume in `/etc/crypttab`, for example:

```
# <target name>   <source device>   <key file>   <options>
cryptfs           /dev/sdd          none         luks
```

This entry causes the operating system to prompt you to enter the passphrase at boot time.

Having created an encrypted volume and its device mapping, you can configure and use it in the same way as you would a physical storage device. For example, you can configure it as an LVM physical volume, file system, swap partition, Automatic Storage Management (ASM) disk, or raw device. For example, you would create an entry in the `/etc/fstab` to mount the mapped device (`/dev/mapper/cryptsfs`), not the physical device (`/dev/sdd`).

To verify the status of an encrypted volume, use the following command:

```
# cryptsetup status cryptfs
/dev/mapper/cryptfs is active.
type: LUKS1
cipher:  aes-cbs-essiv:sha256
keysize: 256 bits
device:  /dev/xvdd1
offset:  4096 sectors
size:    6309386 sectors
mode:    read/write
```

Should you need to remove the device mapping, unmount any file system that the encrypted volume contains, and run the following command:

```
# cryptsetup luksClose /dev/mapper/cryptfs
```

For more information, see the `crypsetup(8)` and `crypttab(5)` manual pages.

19.6 SSD Configuration Recommendations for btrfs, ext4, and swap

When partitioning an SSD, align primary and logical partitions on one-megabyte (1048576 bytes) boundaries. If partitions, file system blocks, or RAID stripes are incorrectly aligned and overlap the boundaries of the underlying storage's pages, which are usually either 4 KB or 8 KB in size, the device controller has to modify twice as many pages than if correct alignment is used.

For btrfs and ext4 file systems, specifying the `discard` option with `mount` sends discard (TRIM) commands to an underlying SSD whenever blocks are freed. This option can extend the working life of the device but it has a negative impact on performance, even for SSDs that support queued discards. The recommended alternative is to use the `fstrim` command to discard empty blocks that the file system is not using, especially before reinstalling the operating system or before creating a new file system on an SSD. Schedule `fstrim` to run when it will have minimal impact on system performance. You can also apply `fstrim` to a specific range of blocks rather than the whole file system.

> **Note**
>
> Using a minimal journal size of 1024 file-system blocks for ext4 on an SSD improves performance. However, it is not recommended that you disable journalling altogether as it improves the robustness of the file system.

Btrfs automatically enables SSD optimization for a device if the value of `/sys/block/device/queue/rotational` is 0. If btrfs does not detect a device as being an SSD, you can enable SSD optimization by specifying the `ssd` option to `mount`.

> **Note**
>
> By default, btrfs enables SSD optimization for Xen Virtual Devices (XVD) because the value of `rotational` for these devices is 0. To disable SSD optimization, specify the `nossd` option to `mount`.

> Setting the `ssd` option does not imply that `discard` is also set.

If you configure swap files or partitions on an SSD, reduce the tendency of the kernel to perform anticipatory writes to swap, which is controlled by the value of the `vm.swappiness` kernel parameter and displayed as `/proc/sys/vm/swappiness`. The value of `vm.swappiness` can be in the range 0 to 100, where a higher value implies a greater propensity to write to swap. The default value is 60. The suggested value when swap has been configured on SSD is 1. You can use the following commands to change the value:

```
# echo "vm.swappiness = 1" >> /etc/sysctl.conf
# sysctl -p
...
vm.swappiness = 1
```

19.7 About Linux-IO Storage Configuration

Oracle Linux 7 with both UEK R3 and RHCK uses the Linux-IO Target (LIO) to provide the block-storage SCSI target for FCoE, iSCSI, and Mellanox InfiniBand (iSER and SRP). You can manage LIO by using the `targetcli` shell provided in the `targetcli` package.

Fibre Channel over Ethernet (FCoE) encapsulates Fibre Channel packets in Ethernet frames, which allows them to be sent over Ethernet networks. To configure FCoE storage, you also need to install the `fcoe-utils` package, which provides the `fcoemon` service and the `fcoeadm` command.

The Internet Small Computer System Interface (iSCSI) is an IP-based standard for connecting storage devices. iSCSI encapsulates SCSI commands in IP network packets, which allows data transfer over long distances and sharing of storage by client systems. As iSCSI uses the existing IP infrastructure, it does not require the purchase and installation of fiber-optic cabling and interface adapters that are needed to implement Fibre Channel (FC) storage area networks.

A client system (*iSCSI initiator*) accesses the storage server (*iSCSI target*) over an IP network. To an iSCSI initiator, the storage appears to be locally attached.

An iSCSI target is typically a dedicated, network-connected storage device but it can also be a general-purpose computer.

Figure 19.1 shows a simple network where several iSCSI initiators are able to access the shared storage that is attached to an iSCSI target.

Figure 19.1 iSCSI Initiators and an iSCSI Target Connected via an IP-based Network
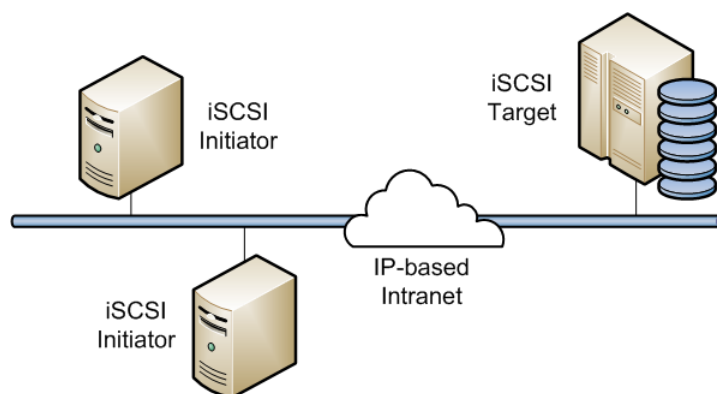
A hardware-based iSCSI initiator uses a dedicated iSCSI HBA. Oracle Linux supports iSCSI initiator functionality in software. The kernel-resident device driver uses the existing network interface card (NIC)

and network stack to emulate a hardware iSCSI initiator. As the iSCSI initiator functionality is not available at the level of the system BIOS, you cannot boot an Oracle Linux system from iSCSI storage .

To improve performance, some network cards implement TCP/IP Offload Engines (TOE) that can create a TCP frame for the iSCSI packet in hardware. Oracle Linux does not support TOE, although suitable drivers may be available directly from some card vendors.

For more information about LIO, see http://linux-iscsi.org/wiki/Main_Page.

19.7.1 Configuring an iSCSI Target

To set up a simple iSCSI target on an Oracle Linux system:

1. Run the `targetcli` shell:

```
# targetcli
targetcli shell version 2.1.fb31
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.
```

   List the object hierarchy, which is initially empty:

```
/> ls
o- / ................................................................ [...]
  o- backstores ..................................................... [...]
  | o- block ......................................... [Storage Objects: 0]
  | o- fileio ........................................ [Storage Objects: 0]
  | o- pscsi ......................................... [Storage Objects: 0]
  | o- ramdisk ....................................... [Storage Objects: 0]
  o- iscsi .................................................. [Targets: 0]
  o- loopback ............................................... [Targets: 0]
```

2. Change to the `/backstores/block` directory and create a block storage object for the disk partitions that you want to provide as LUNs, for example:

```
/> cd /backstores/block
/backstores/block> create name=LUN_0 dev=/dev/sdb
Created block storage object LUN_0 using /dev/sdb.
/backstores/block> create name=LUN_1 dev=/dev/sdc
Created block storage object LUN_1 using /dev/sdc.
```

   The names that you assign to the storage objects are arbitrary.

3. Change to the `/iscsi` directory and create an iSCSI target:

```
/> cd /iscsi
/iscsi> create
Created target iqn.2013-01.com.mydom.host01.x8664:sn.ef8e14f87344.
Created TPG 1.
```

   List the target portal group (TPG) hierarchy, which is initially empty:

```
/iscsi> ls
o- iscsi ....................................................... [Targets: 1]
  o- iqn.2013-01.com.mydom.host01.x8664:sn.ef8e14f87344 ............... [TPGs: 1]
    o- tpg1 .............................................. [no-gen-acls, no-auth]
      o- acls ................................................... [ACLs: 0]
      o- luns ................................................... [LUNs: 0]
      o- portals ............................................. [Portals: 0]
```

4. Change to the `luns` subdirectory of the TPG directory hierarchy and add the LUNs to the target portal group:

```
/iscsi> cd iqn.2013-01.com.mydom.host01.x8664:sn.ef8e14f87344/tpg1/luns
/iscsi/iqn.20...344/tpg1/luns> create /backstores/block/LUN_0
Created LUN 0.
/iscsi/iqn.20...344/tpg1/luns> create /backstores/block/LUN_1
Created LUN 1.
```

5. Change to the portals subdirectory of the TPG directory hierarchy and specify the IP address and port of the iSCSI endpoint:

```
/iscsi/iqn.20...344/tpg1/luns> cd ../portals
/iscsi/iqn.20.../tpg1/portals> create 10.150.30.72 3260
Using default IP port 3260
Created network portal 10.150.30.72:3260.
```

If you omit the port number, the default value is 3260.

List the object hierarchy, which now shows the configured block storage objects and TPG:

```
/iscsi/iqn.20.../tpg1/portals> ls /
o- / ................................................................ [...]
  o- backstores .................................................... [...]
  | o- block ......................................... [Storage Objects: 1]
  | | o- LUN_0 ....................... [/dev/sdb (10.0GiB) write-thru activated]
  | | o- LUN_1 ....................... [/dev/sdc (10.0GiB) write-thru activated]
  | o- fileio ........................................ [Storage Objects: 0]
  | o- pscsi ......................................... [Storage Objects: 0]
  | o- ramdisk ....................................... [Storage Objects: 0]
  o- iscsi ................................................... [Targets: 1]
  | o- iqn.2013-01.com.mydom.host01.x8664:sn.ef8e14f87344 ............. [TPGs: 1]
  |   o- tpg1 ......................................... [no-gen-acls, no-auth]
  |     o- acls .................................................... [ACLs: 0]
  |     o- luns .................................................... [LUNs: 1]
  |     | o- lun0 ..................................... [block/LUN_0 (/dev/sdb)]
  |     | o- lun1 ..................................... [block/LUN_1 (/dev/sdc)]
  |     o- portals ............................................... [Portals: 1]
  |       o- 10.150.30.72:3260 .......................................... [OK]
  o- loopback ................................................ [Targets: 0]
```

6. Configure the access rights for logins by initiators. For example, to configure demonstration mode that does not require authentication, change to the TGP directory and set the values of the authentication and demo_mode_write_protect attributes to 0 and generate_node_acls cache_dynamic_acls to 1:

```
/iscsi/iqn.20.../tpg1/portals> cd ..
/iscsi/iqn.20...14f87344/tpg1> set attribute authentication=0 demo_mode_write_protect=0 \
generate_node_acls=1 cache_dynamic_acls=1
Parameter authentication is now '0'.
Parameter demo_mode_write_protect is now '0'.
Parameter generate_node_acls is now '1'.
Parameter cache_dynamic_acls is now '1'.
```

⚠️ **Caution**

Demonstration mode is inherently insecure. For information about configuring secure authentication modes, see http://linux-iscsi.org/wiki/ISCSI#Define_access_rights.

7. Change to the root directory and save the configuration so that it persists across reboots of the system:

```
/iscsi/iqn.20...14f87344/tpg1> cd /
/> saveconfig
Last 10 configs saved in /etc/target/backup.
Configuration saved to /etc/target/saveconfig.json
```

targetcli saves the current configuration to the JSON-format file `/etc/target/`
`saveconfig.json`.

For more information, see the `targetcli(8)` manual page.

19.7.2 Configuring an iSCSI Initiator

To configure an Oracle Linux system as an iSCSI initiator:

1. Install the `iscsi-initiator-utils` package:

    ```
    # yum install iscsi-initiator-utils
    ```

2. Use the *SendTargets* discovery method to discover the iSCSI targets at a specified IP address:

    ```
    # iscsiadm -m discovery -t sendtargets -p 10.150.30.72
    10.150.30.72:3260,1 iqn.2013-01.com.mydom.host01.x8664:sn.ef8e14f87344
    ```

    > **Note**
    >
    > An alternate discovery method is Internet Storage Name Service (iSNS).

    The command also starts the `iscsid` service if it is not already running.

    The following command displays information about the targets that is now stored in the discovery
    database:

    ```
    # iscsiadm -m discoverydb -t st -p 10.150.30.72
    # BEGIN RECORD 6.2.0.873-14
    discovery.startup = manual
    discovery.type = sendtargets
    discovery.sendtargets.address = 10.150.30.72
    discovery.sendtargets.port = 3260
    discovery.sendtargets.auth.authmethod = None
    discovery.sendtargets.auth.username = <empty>
    discovery.sendtargets.auth.password = <empty>
    discovery.sendtargets.auth.username_in = <empty>
    discovery.sendtargets.auth.password_in = <empty>
    discovery.sendtargets.timeo.login_timeout = 15
    discovery.sendtargets.use_discoveryd = No
    discovery.sendtargets.discoveryd_poll_inval = 30
    discovery.sendtargets.reopen_max = 5
    discovery.sendtargets.timeo.auth_timeout = 45
    discovery.sendtargets.timeo.active_timeout = 30
    discovery.sendtargets.iscsi.MaxRecvDataSegmentLength = 32768
    # END RECORD
    ```

3. Establish a session and log in to a specific target:

    ```
    # iscsiadm -m node -T iqn.2013-01.com.mydom.host01.x8664:sn.ef8e14f87344 \
      -p 10.150.30.72:3260 -l
    Login to [iface: default, target: iqn.2003-01.org.linux-iscsi.localhost.x8664:
    sn.ef8e14f87344, portal: 10.150.30.72,3260] successful.
    ```

4. Verify that the session is active, and display the available LUNs:

    ```
    # iscsiadm -m session -P 3
    iSCSI Transport Class version 2.0-870
    version 6.2.0.873-14
    Target: iqn.2003-01.com.mydom.host01.x8664:sn.ef8e14f87344 (non-flash)
     Current Portal: 10.0.0.2:3260,1
     Persistent Portal: 10.0.0.2:3260,1
    ```

```
**********
Interface:
**********
Iface Name: default
Iface Transport: tcp
Iface Initiatorname: iqn.1994-05.com.mydom:ed7021225d52
Iface IPaddress: 10.0.0.2
Iface HWaddress: <empty>
Iface Netdev: <empty>
SID: 5
iSCSI Connection State: LOGGED IN
iSCSI Session State: LOGGED_IN
Internal iscsid Session State: NO CHANGE
.
.
.
***********************
Attached SCSI devices:
***********************
Host Number: 8 State: running
scsi8 Channel 00 Id 0 Lun: 0
 Attached scsi disk sdb  State: running
scsi8 Channel 00 Id 0 Lun: 1
 Attached scsi disk sdc  State: running
```

The LUNs are represented as SCSI block devices (sd*) in the local /dev directory, for example:

```
# fdisk -l | grep /dev/sd[bc]
Disk /dev/sdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/sdc: 10.7 GB, 10737418240 bytes, 20971520 sectors
```

To distinguish between target LUNs, examine their paths under /dev/disk/by-path:

```
# ls -l /dev/disk/by-path/
lrwxrwxrwx  1 root root  9 May 15 21:05
  ip-10.150.30.72:3260-iscsi-iqn.2013-01.com.mydom.host01.x8664:
  sn.ef8e14f87344-lun-0 -> ../../sdb
lrwxrwxrwx 1 root root  9 May 15 21:05
  ip-10.150.30.72:3260-iscsi-iqn.2013-01.com.mydom.host01.x8664:
  sn.ef8e14f87344-lun-1 -> ../../sdc
```

You can view the initialization messages for the LUNs in the /var/log/messages file, for example:

```
# grep sdb /var/log/messages
...
May 18 14:19:36 localhost kernel: [12079.963376] sd 8:0:0:0: [sdb] Attached SCSI disk
...
```

You can configure and use a LUN in the same way as you would any other physical storage device. For example, you can configure it as an LVM physical volume, file system, swap partition, Automatic Storage Management (ASM) disk, or raw device.

Specify the _netdev option when creating mount entries for iSCSI LUNs in /etc/fstab, for example:

```
UUID=084591f8-6b8b-c857-f002-ecf8a3b387f3    /iscsi_mount_point    ext4    _netdev   0  0
```

This option indicates the file system resides on a device that requires network access, and prevents the system from attempting to mount the file system until the network has been enabled.

Note

Specify an iSCSI LUN in /etc/fstab by using UUID=*UUID* rather than the device path. A device path can change after re-connecting the storage or

215

> rebooting the system. You can use the `blkid` command to display the `UUID` of a block device.
>
> Any discovered LUNs remain available across reboots provided that the target continues to serve those LUNs and you do not log the system off the target.

For more information, see the `iscsiadm(8)` and `iscsid(8)` manual pages.

19.7.3 Updating the Discovery Database

If the LUNs that are available on an iSCSI target change, you can use the `iscsiadm` command on an iSCSI initiator to update the entries in its discovery database. The following example assume that the target supports the SendTargets discovery method

To add new records that are not currently in the database:

```
# iscsiadm --mode discoverydb -type st -p 10.150.30.72 -o new --discover
```

To update existing records in the database:

```
# iscsiadm -m discoverydb -t st -p 10.150.30.72 -o update --discover
```

To delete records from the database that are no longer supported by the target:

```
# iscsiadm -m discoverydb -t st -p 10.150.30.72 -o delete --discover
```

For more information, see the `iscsiadm(8)` manual page.

19.8 About Device Multipathing

Multiple paths to storage devices can provide connection redundancy, failover capability, load balancing, and improved performance. Device-Mapper Multipath (DM-Multipath) is a multipathing tool that allows you to represent multiple I/O paths between a server and a storage device as a single path.

You would be most likely to configure multipathing with a system that can access storage on a Fibre Channel-based storage area network (SAN). You can also use multipathing on an iSCSI initiator if redundant network connections exist between the initiator and the target.

Figure 19.2 shows a simple DM-Multipath configuration where two I/O paths are configured between a server and a disk on a SAN-attached storage array:

- Between host bus adapter `hba1` on the server and controller `ctrl1` on the storage array.

- Between host bus adapter `hba2` on the server and controller `ctrl2` on the storage array.

Figure 19.2 DM-Multipath Mapping of Two Paths to a Disk over a SAN
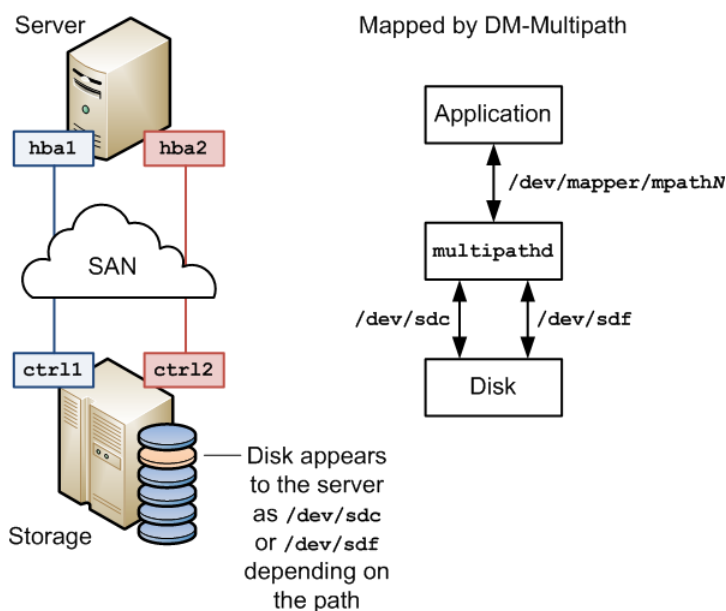
Without DM-Multipath, the system treats each path as being separate even though it connects the server to the same storage device. DM-Multipath creates a single multipath device, `/dev/mapper/mpathN`, that subsumes the underlying devices, `/dev/sdc` and `/dev/sdf`.

You can configure the multipathing service (`multipathd`) to handle I/O from and to a multipathed device in one of the following ways:

Active/Active	I/O is distributed across all available paths, either by round-robin assignment or dynamic load-balancing.
Active/Passive (standby failover)	I/O uses only one path. If the active path fails, DM-Multipath switches I/O to a standby path. This is the default configuration.

> **Note**
>
> DM-Multipath can provide failover in the case of path failure, such as in a SAN fabric. Disk media failure must be handled by using either a software or hardware RAID solution.

19.8.1 Configuring Multipathing

The procedure in this section demonstrates how to set up a simple multipath configuration.

To configure multipathing on a server with access to SAN-attached storage:

1. Install the `device-mapper-multipath` package:

   ```
   # yum install device-mapper-multipath
   ```

2. You can now choose one of two configuration paths:

   - To set up a basic standby failover configuration without editing the `/etc/multipath.conf` configuration file, enter the following command:

     ```
     # mpathconf --enable --with_multipathd y
     ```

This command also starts the `multipathd` service and configures the service to start after system reboots.

Skip the remaining steps of this procedure.

- To edit `/etc/multipath.conf` and set up a more complex configuration such as active/active, follow the remaining steps in this procedure.

3. Initialize the `/etc/multipath.conf` file:

```
# mpathconf --enable
```

4. Edit `/etc/multipath.conf` and define `defaults`, `blacklist`, `blacklist_exceptions`, `multipaths`, and `devices` sections as required, for example:

```
defaults {
    udev_dir              /dev
    polling_interval      10
    path_selector         "round-robin 0"
    path_grouping_policy  multibus
    getuid_callout        "/lib/udev/scsi_id --whitelisted --device=/dev/%n"
    prio                  alua
    path_checker          readsector0
    rr_min_io             100
    max_fds               8192
    rr_weight             priorities
    failback              immediate
    no_path_retry         fail
    user_friendly_names   yes
}

blacklist {
    # Blacklist by WWID
    wwid "*"

    # Blacklist by device name
    devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st)[0-9]*"

    # Blacklist by device type
    device {
      vendor    "COMPAQ  "
      product   "HSV110 (C)COMPAQ"
    }
}

blacklist_exceptions {
    wwid "3600508b4000156d700012000000b0000"
    wwid "360000970000292602744533032443941"
}

multipaths {
    multipath {
        wwid                  3600508b4000156d700012000000b0000
        alias                 blue
        path_grouping_policy  multibus
        path_checker          readsector0
        path_selector         "round-robin 0"
        failback              manual
        rr_weight             priorities
        no_path_retry         5
    }
    multipath {
        wwid                  360000970000292602744533032443941
        alias                 green
```

```
    }
}

devices {
    device {
        vendor                "SUN"
        product               "(StorEdge 3510|T4"
        path_grouping_policy  multibus
        getuid_callout        "/sbin/scsi_id --whitelisted --device=/dev/%n"
        path_selector         "round-robin 0"
        features              "0"
        hardware_handler      "0"
        path_checker          directio
        prio                  const
        rr_weight             uniform
        rr_min_io             1000
    }
}
```

The sections have the following purposes:

defaults   Defines default multipath settings, which can be overridden
                              by settings in the devices section, and which in turn can be
                              overridden by settings in the multipaths section.

blacklist   Defines devices that are excluded from multipath topology
                              discovery. Blacklisted devices cannot subsumed by a multipath
                              device.

                              The example shows the three ways that you can use to exclude
                              devices: by WWID (wwid), by device name (devnode), and by
                              device type (device).

blacklist_exceptions   Defines devices that are included in multipath topology discovery,
                              even if the devices are implicitly or explicitly listed in the blacklist
                              section.

multipaths   Defines settings for a multipath device that is identified by its WWID.

                              The alias attribute specifies the name of the multipath device as it
                              will appear in /dev/mapper instead of a name based on either the
                              WWID or the multipath group number.

                              To obtain the WWID of a SCSI device, use the scsi_id command:

                              # **scsi_id --whitelisted --replace-whitespace --device=**_device_name_

devices   Defines settings for individual types of storage controller. Each
                              controller type is identified by the vendor, product, and optional
                              revision settings, which must match the information in sysfs for
                              the device.

                              You can find details of the storage arrays that DM-Multipath
                              supports and their default configuration values in /usr/
                              share/doc/device-mapper-multipath-_version_/
                              multipath.conf.defaults, which you can use as the basis for
                              entries in /etc/multipath.conf.

To add a storage device that DM-Multipath does not list as being supported, obtain the vendor, product, and revision information from the `vendor`, `model`, and `rev` files under `/sys/block/`*`device_name`*`/device`.

The following entries in `/etc/multipath.conf` would be appropriate for setting up active/passive multipathing to an iSCSI LUN with the specified WWID.

```
defaults {
    user_friendly_names     yes
    getuid_callout          "/bin/scsi_id --whitelisted --replace-whitespace --device=/dev/%n"
}

multipaths {
    multipath {
        wwid 3600009700002926027445330303703730
    }
}
```

In this standby failover configuration, I/O continues through a remaining active network interface if a network interfaces fails on the iSCSI initiator.

For more information about configuring entries in `/etc/multipath.conf`, refer to the `multipath.conf(5)` manual page.

5. Start the `multipathd` service and configure the service to start after system reboots:

```
# systemctl start multipathd
# systemctl enable multipathd
```

Multipath devices are identified in `/dev/mapper` by their World Wide Identifier (WWID), which is globally unique. Alternatively, if you set the value of `user_friendly_names` to `yes` in the `defaults` section of `/etc/multipath.conf` or by specifying the `--user_friendly_names n` option to `mpathconf`, the device is named `mpath`*`N`* where *`N`* is the multipath group number. An `alias` attribute in the `multipaths` section of `/etc/multipath.conf` specifies the name of the multipath device instead of a name based on either the WWID or the multipath group number.

You can use the multipath device in `/dev/mapper` to reference the storage in the same way as you would any other physical storage device. For example, you can configure it as an LVM physical volume, file system, swap partition, Automatic Storage Management (ASM) disk, or raw device.

To display the status of DM-Multipath, use the `mpathconf` command, for example:

```
# mpathconf
multipath is enabled
find_multipaths is enabled
user_friendly_names is enabled
dm_multipath modules is loaded
multipathd is running
```

To display the current multipath configuration, specify the `-ll` option to the `multipath` command, for example:

```
# multipath -ll
mpath1(3600009700002926027445330303703730) dm-0 SUN,(StorEdge 3510|T4
size=20G features='0' hwhandler='0' wp=rw
|-+- policy='round-robin 0' prio=1 status=active
| `- 5:0:0:2 sdb 8:16    active ready running
`-+- policy='round-robin 0' prio=1 status=active
  `- 5:0:0:3 sdc 8:32    active ready running
```

In this example, `/dev/mapper/mpath1` subsumes two paths (`/dev/sdb` and `/dev/sdc`) to 20 GB of storage in an active/active configuration using round-robin I/O path selection. The WWID that identifies the storage is `3600009700002926027445330303030730` and the name of the multipath device under `sysfs` is `dm-0`.

If you edit `/etc/multipath.conf`, restart the `multipathd` service to make it re-read the file:

```
# systemctl restart multipathd
```

For more information, see the `mpathconf(8)`, `multipath(8)`, `multipathd(8)`, `multipath.conf(5)`, and `scsi_id(8)` manual pages.

Chapter 20 File System Administration

Table of Contents

This chapter describes how to create, mount, check, and repair file systems, how to configure Access Control Lists, how to configure and manage disk quotas.

20.1 Making File Systems

The `mkfs` command build a file system on a block device:

```
# mkfs [options] device
```

`mkfs` is a front end for builder utilities in `/sbin` such as `mkfs.ext4`. You can use either the `mkfs` command with the `-t fstype` option or the builder utility to specify the type of file system to build. For example, the following commands are equivalent ways of creating an `ext4` file system with the label `Projects` on the device `/dev/sdb1`:

```
# mkfs -t ext4 -L Projects /dev/sdb1
# mkfs.ext4 -L Projects /dev/sdb1
```

If you do not specify the file system type to `makefs`, it creates an `ext2` file system.

To display the type of a file system, use the `blkid` command:

```
# blkid /dev/sdb1
/dev/sdb1: UUID="ad8113d7-b279-4da8-b6e4-cfba045f66ff" TYPE="ext4" LABEL="Projects"
```

The `blkid` command also display information about the device such as its UUID and label.

Each file system type supports a number of features that you can enable or disable by specifying additional options to `mkfs` or the build utility. For example, you can use the `-J` option to specify the size and location of the journal used by the `ext3` and `ext4` file system types.

For more information, see the `blkid(8)`, `mkfs(8)`, and `mkfs.fstype(8)` manual pages.

20.2 Mounting File Systems

To access a file system's contents, you must attach its block device to a mount point in the directory hierarchy. You can use the `mkdir` command to create a directory for use as a mount point, for example:

```
# mkdir /var/projects
```

You can use an existing directory as a mount point, but its contents are hidden until you unmount the overlying file system.

The `mount` command attaches the device containing the file system to the mount point:

```
# mount [options] device mount_point
```

You can specify the device by its name, UUID, or label. For example, the following commands are equivalent ways of mounting the file system on the block device `/dev/sdb1`:

```
# mount /dev/sdb1 /var/projects
# mount UUID="ad8113d7-b279-4da8-b6e4-cfba045f66ff" /var/projects
# mount LABEL="Projects" /var/projects
```

If you do not specify any arguments, `mount` displays all file systems that the system currently has mounted, for example:

```
# mount
/dev/mapper/vg_host01-lv_root on / type ext4 (rw)
...
```

In this example, the LVM logical volume `/dev/mapper/vg_host01-lv_root` is mounted on `/`. The file system type is `ext4` and is mounted for both reading and writing. (You can also use the command `cat /proc/mounts` to display information about mounted file systems.)

The `df` command displays information about home much space remains on mounted file systems, for example:

```
# df -h
Filesystem                     Size  Used Avail Use% Mounted on
/dev/mapper/vg_host01-lv_root  36G   12G   22G  36% /
...
```

You can use the `-B` (bind) option to the `mount` command to attach a block device at multiple mount points. You can also remount part of a directory hierarchy, which need not be a complete file system, somewhere else. For example, the following command mounts `/var/projects/project1` on `/mnt`:

```
# mount -B /var/projects/project1 /mnt
```

Each directory hierarchy acts as a mirror of the other. The same files are accessible in either location, although any submounts are not replicated. These mirrors do not provide data redundancy.

You can also mount a file over another file, for example:

```
# touch /mnt/foo
# mount -B /etc/hosts /mnt/foo
```

In this example, `/etc/hosts` and `/mnt/foo` represent the same file. The existing file that acts as a mount point is not accessible until you unmount the overlying file.

The `-B` option does not recursively attach any submounts below a directory hierarchy. To include submounts in the mirror, use the `-R` (recursive bind) option instead.

When you use `-B` or `-R`, the file system `mount` options remain the same as those for the original mount point. To modify, the mount options, use a separate remount command, for example:

```
# mount -o remount,ro /mnt/foo
```

You can mark the submounts below a mount point as being shared, private, or slave:

mount --make-shared
mount_point
Any mounts or unmounts below the specified mount point propagate to any mirrors that you create, and this mount hierarchy reflects mounts or unmount changes that you make to other mirrors.

mount --make-private
mount_point
Any mounts or unmounts below the specified mount point do not propagate to other mirrors, nor does this mount hierarchy reflect mounts or unmount changes that you make to other mirrors.

mount --make-slave
mount_point
Any mounts or unmounts below the specified mount point do not propagate to other mirrors, but this mount hierarchy does reflect mounts or unmount changes that you make to other mirrors.

To prevent a mount from being mirrored by using the `-B` or `-R` options, mark its mount point as being unbindable:

```
# mount --make-unbindable mount_point
```

To move a mounted file system, directory hierarchy, or file between mount points, use the `-M` option, for example:

```
# touch /mnt/foo
# mount -M /mnt/foo /mnt/bar
```

To unmount a file system, use the `umount` command, for example:

```
# umount /var/projects
```

Alternatively, you can specify the block device provided that it is mounted on only one mount point.

For more information, see the `mount(8)` and `umount(8)` manual pages.

20.2.1 About Mount Options

To modify the behavior of `mount`, use the `-o` flag followed by a comma-separated list of options or specify the options in the `/etc/fstab` file. The following are some of the options that are available:

auto   Allows the file system to be mounted automatically by using the `mount -a` command.

exec   Allows the execution of any binary files located in the file system.

loop   Uses a loop device (`/dev/loop*`) to mount a file that contains a file system image. See Section 20.5, "Mounting a File Containing a File System Image", Section 20.6, "Creating a File System on a File", and the `losetup(8)` manual page.

Note

The default number of available loop devices is 8. You can use the kernel boot parameter `max_loop=`*N* to configure up to 255 devices. Alternatively, add the following entry to `/etc/modprobe.conf`:

```
options loop max_loop=N
```

where *N* is the number of loop devices that you require (from 0 to 255), and reboot the system.

noauto   Disallows the file system from being mounted automatically by using `mount -a`.

noexec   Disallows the execution of any binary files located in the file system.

nouser   Disallows any user other than `root` from mounting or unmounting the file system.

remount   Remounts the file system if it is already mounted. You would usually combine this option with another option such as `ro` or `rw` to change the behavior of a mounted file system.

ro   Mounts a file system as read-only.

rw   Mounts a file system for reading and writing.

user   Allows any user to mount or unmount the file system.

For example, mount `/dev/sdd1` as `/test` with read-only access and only root permitted to mount or unmount the file system:

```
# mount -o nouser,ro /dev/sdd1 /test
```

Mount an ISO image file on `/mount/cdrom` with read-only access by using the loop device:

```
# mount -o ro,loop ./OracleLinux-R6-U1-Server-x86_64-dvd.iso /media/cdrom
```

Remount the `/test` file system with both read and write access, but do not permit the execution of any binary files that are located in the file system:

```
# mount -o remount,rw,noexec /test
```

20.3 About the File System Mount Table

The `/etc/fstab` file contains the file system mount table, and provides all the information that the `mount` command needs to mount block devices or to implement binding of mounts. If you add a file system, create the appropriate entry in `/etc/fstab` to ensure that the file system is mounted at boot time. The following are sample entries from `/etc/fstab`:

```
/dev/sda1        /boot   ext4     defaults  1 2
/dev/sda2        /       ext4     defaults  1 1
/dev/sda3        swap    swap     defaults  0 0
```

The first field is the device to mount specified by the device name, UUID, or device label, or the specification of a remote file system. A UUID or device label is preferable to a device name if the device name could change, for example:

```
LABEL=Projects     /var/projects  ext4  defaults  1 2
```

The second field is either the mount point for a file system or `swap` to indicate a swap partition.

The third field is the file system type, for example `ext4` or `swap`.

The fourth field specifies any mount options.

The fifth column is used by the `dump` command. A value of 1 means dump the file system; 0 means the file system does not need to be dumped.

The sixth column is used by the file system checker, `fsck`, to determine in which order to perform file system checks at boot time. The value should be 1 for the root file system, 2 for other file systems. A value of 0 skips checking, as is appropriate for swap, file systems that are not mounted at boot time, or for binding of existing mounts.

For bind mounts, only the first four fields are specified, for example:

```
path          mount_point    none    bind
```

The first field specifies the path of the file system, directory hierarchy, or file that is to be mounted on the mount point specified by the second field. The mount point must be a file if the path specifies a file; otherwise, it must be a directory. The third and fourth fields are specified as `none` and `bind`.

For more information, see the `fstab(5)` manual page.

20.4 Configuring the Automounter

The automounter mounts file systems when they are accessed, rather than maintaining connections for those mounts at all times. When a file system becomes inactive for more than a certain period of time, the automounter unmounts it. Using automounting frees up system resources and improves system performance.

The automounter consists of two components: the `autofs` kernel module and the `automount` user-space daemon.

To configure a system to use automounting:

1. Install the `autofs` package and any other packages that are required to support remote file systems:

   ```
   # yum install autofs
   ```

2. Edit the `/etc/auto.master` configuration file to define map entries. Each map entry specifies a mount point and a map file that contains definitions of the remote file systems that can be mounted, for example:

   ```
   /-          /etc/auto.direct
   /misc       /etc/auto.misc
   /net        -hosts
   ```

   Here, the `/-`, `/misc`, and `/net` entries are examples of a direct map, an indirect map, and a host map respectively. Direct map entries always specify `/-` as the mount point. Host maps always specify the keyword `-hosts` instead of a map file.

   A direct map contains definitions of directories that are automounted at the specified absolute path. In the example, the `auto.direct` map file might contain an entry such as:

   ```
   /usr/man    -fstype=nfs,ro,soft            host01:/usr/man
   ```

   This entry mounts the file system `/usr/man` exported by `host01` using the options `ro` and `soft`, and creates the `/usr/man` mount point if it does not already exist. If the mount point already exists , the mounted file system hides any existing files that it contains.

   As the default file system type is NFS, the previous example can be shortened to read:

   ```
   /usr/man    -ro,soft                       host01:/usr/man
   ```

   An indirect map contains definitions of directories (*keys*) that are automounted relative to the mount point (`/misc`) specified in `/etc/auto.master`. In the example, the `/etc/auto.misc` map file might contain entries such as the following:

   ```
   xyz         -ro,soft                       host01:/xyz
   cd          -fstype=iso9600,ro,nosuid,nodev        :/dev/cdrom
   abc         -fstype=ext3                           :/dev/hda1
   fenetres    -fstype=cifs,credentials=credfile      ://fenetres/c
   ```

The `/misc` directory must already exist, but the automounter creates a mount point for the keys `xyz`, `cd` , and so on if they does not already exist, and removes them when it unmounts the file system. For example, entering a command such as `ls /misc/xyz` causes the automounter to the mount the `/xyz` directory exported by `host01` as `/misc/xyz`.

The `cd` and `abc` entries mount local file systems: an ISO image from the CD-ROM drive on `/misc/cd` and an ext3 file system from `/dev/hda1` on `/misc/abc`. The `fenetres` entry mounts a Samba share as `/misc/fenetres`.

If a host map entry exists and a command references an NFS server by name relative to the mount point (`/net`), the automounter mounts all directories that the server exports below a subdirectory of the mount point named for the server. For example, the command `cd /net/host03` causes the automounter to mount all exports from `host03` below the `/net/host03` directory. By default, the automounter uses the mount options `nosuid,nodev,intr` options unless you override the options in the host map entry, for example:

```
/net        -hosts    -suid,dev,nointr
```

Note

The name of the NFS server must be resolvable to an IP address in DNS or in the `/etc/hosts` file.

For more information, including details of using maps with NIS, NIS+, and LDAP, see the `hosts.master(5)` manual page.

3. Start the `autofs` service, and configure the service to start following a system reboot:

```
# systemctl stat autofs
# systemctl enable autofs
```

You can configure various settings for `autofs` in `/etc/sysconfig/autofs`, such as the idle timeout value after which a file system is automatically unmounted.

If you modify `/etc/auto.master` or `/etc/sysconfig/autofs`, restart the `autofs` service to make it re-read these files:

```
# systemctl restart autofs
```

For more information, see the `automount(8)`, `autofs(5)`, and `auto.master(5)` manual pages.

20.5 Mounting a File Containing a File System Image

A loop device allows you to access a file as a block device. For example, to mount a file that contains a DVD ISO image on the directory mount point `/ISO`:

```
# mount -t iso9660 -o ro,loop /var/ISO_files/V33411-01.iso /ISO
```

If required, create a permanent entry for the file system in `/etc/fstab`:

```
/var/ISO_files/V33411-01.iso            /ISO      iso9660    ro,loop    0 0
```

20.6 Creating a File System on a File

To create a file system on a file within another file system:

1. Create an empty file of the required size, for example:

```
# dd if=/dev/zero of=/fsfile bs=1024 count=1000000
1000000+0 records in
1000000+0 records out
1024000000 bytes (1.0 GB) copied, 8.44173 s, 121 MB/s
```

2. Create a file system on the file:

```
# mkfs.ext4 -F /fsfile
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
62592 inodes, 250000 blocks
12500 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=260046848
8 block groups
32768 blocks per group, 32768 fragments per group
7824 inodes per group
Superblock backups stored on blocks:
 32768, 98304, 163840, 229376

Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 33 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

3. Mount the file as a file system by using a loop device:

```
# mount -o loop /fsfile /mnt
```

The file appears as a normal file system:

```
# mount
...
/fsfile on /mnt type ext4 (rw,loop=/dev/loop0)
# df -h
Filesystem          Size  Used Avail Use% Mounted on
...
/fsfile             962M   18M  896M   2% /mnt
```

If required, create a permanent entry for the file system in `/etc/fstab`:

```
/fsfile          /mnt      ext4    rw,loop     0 0
```

20.7 Checking and Repairing a File System

The `fsck` utility checks and repairs file systems. For file systems other than `/` (root) and `/boot`, `mount` invokes file system checking if more than a certain number of mounts have occurred or more than 180 days have elapsed without checking having being performed. You might want to run `fsck` manually if a file system has not been checked for several months.

> **Warning**
>
> Running `fsck` on a mounted file system can corrupt the file system and cause data loss.

To check and repair a file system:

1. Unmount the file system:

   ```
   # umount filesystem
   ```

2. Use the `fsck` command to check the file system:

   ```
   # fsck [-y] filesystem
   ```

   filesystem be a device name, a mount point, or a label or UUID specifier, for example:

   ```
   # fsck UUID=ad8113d7-b279-4da8-b6e4-cfba045f66ff
   ```

   By default, `fsck` prompts you to choose whether it should apply a suggested repair to the file system. If you specify the `-y` option, `fsck` assumes a **yes** response to all such questions.

For the `ext2`, `ext3`, and `ext4` file system types, other commands that are used to perform file system maintenance include `dumpe2fs` and `debugfs`. `dumpe2fs` prints super block and block group information for the file system on a specified device. `debugfs` is an interactive file system debugger that requires expert knowledge of the file system architecture. Similar commands exist for most file system types and also require expert knowledge.

For more information, see the `fsck(8)` manual page.

20.7.1 Changing the Frequency of File System Checking

To change the number of mounts before the system automatically checks the file system for consistency:

```
# tune2fs -c mount_count device
```

where *device* specifies the block device corresponding to the file system.

A *mount_count* of 0 or -1 disables automatic checking based on the number of mounts.

> **Tip**
>
> Specifying a different value of *mount_count* for each file system reduces the probability that the system checks all the file systems at the same time.

To specify the maximum interval between file system checks:

```
# tune2fs -i interval[unit] device
```

The *unit* can be `d`, `w`, or `m` for days, weeks, or months. The default unit is `d` for days. An *interval* of 0 disables checking that is based on the time that has elapsed since the last check. Even if the interval is exceeded, the file system is not checked until it is next mounted.

For more information, see the `tune2fs(8)` manual page.

20.8 About Access Control Lists

POSIX Access Control Lists (ACLs) provide a richer access control model than traditional UNIX Discretionary Access Control (DAC) that sets read, write, and execute permissions for the owner, group, and all other system users. You can configure ACLs that define access rights for more than just a single user or group, and specify rights for programs, processes, files, and directories. If you set a default ACL on a directory, its descendents inherit the same rights automatically. You can use ACLs with btrfs, ext3, ext4, OCFS2, and XFS file systems and with mounted NFS file systems.

An ACL consists of a set of rules that specify how a specific user or group can access the file or directory with which the ACL is associated. A regular ACL entry specifies access information for a single file or

directory. A default ACL entry is set on directories only, and specifies default access information for any file within the directory that does not have an access ACL.

20.8.1 Configuring ACL Support

To enable ACL support:

1. Install the `acl` package:

```
# yum install acl
```

2. Edit `/etc/fstab` and change the entries for the file systems with which you want to use ACLs so that they include the appropriate option that supports ACLs, for example:

```
LABEL=/work        /work        ext4      acl      0 0
```

For mounted Samba shares, use the `cifsacl` option instead of `acl`.

3. Remount the file systems, for example:

```
# mount -o remount /work
```

20.8.2 Setting and Displaying ACLs

To add or modify the ACL rules for file, use the `setfacl` command:

```
# setfacl -m rules file ...
```

The rules take the following forms:

`[d:]u:user[:permissions]`	Sets the access ACL for the user specified by name or user ID. The permissions apply to the owner if a user is not specified.
`[d:]g:group[:permissions]`	Sets the access ACL for a group specified by name or group ID. The permissions apply to the owning group if a group is not specified.
`[d:]m[:][:permissions]`	Sets the effective rights mask, which is the union of all permissions of the owning group and all of the user and group entries.
`[d:]o[:][:permissions]`	Sets the access ACL for other (everyone else to whom no other rule applies).

The permissions are `r`, `w`, and `x` for read, write, and execute as used with `chmod`.

The `d:` prefix is used to apply the rule to the default ACL for a directory.

To display a file's ACL, use the `getfacl` command, for example:

```
# getfacl foofile
# file: foofile
# owner: bob
# group: bob
user::rw-
user::fiona:r--
user::jack:rw-
user::jill:rw-
group::r--
mask::r--
other::r--
```

If extended ACLs are active on a file, the `-l` option to `ls` displays a plus sign (+) after the permissions, for example:

```
# ls -l foofile
-rw-r--r--+ 1 bob bob  105322 Apr 11 11:02 foofile
```

The following are examples of how to set and display ACLs for directories and files.

Grant read access to a file or directory by a user.

```
# setfacl -m u:user:r file
```

Display the name, owner, group, and ACL for a file or directory.

```
# getfacl file
```

Remove write access to a file for all groups and users by modifying the effective rights mask rather than the ACL.

```
# setfacl -m m::rx file
```

The `-x` option removes rules for a user or group.

Remove the rules for a user from the ACL of a file.

```
# setfacl -x u:user file
```

Remove the rules for a group from the ACL of a file.

```
# setfacl -x g:group file
```

The `-b` option removes all extended ACL entries from a file or directory.

```
# setfacl -b file
```

Copy the ACL of file `f1` to file `f2`.

```
# getfacl f1 | setfacl --set-file=- f2
```

Set a default ACL of read and execute access for other on a directory:

```
# setfacl -m d:o:rx directory
```

Promote the ACL settings of a directory to default ACL settings that can be inherited.

```
# getfacl --access directory | setfacl -d -M- directory
```

The `-k` option removes the default ACL from a directory.

```
# setfacl -k directory
```

For more information, see the `acl(5)`, `setfacl(1)`, and `getfacl(1)` manual pages.

20.9 About Disk Quotas

Note

For information about how to configure quotas for the XFS file system, see Section 21.23, "Setting Quotas on an XFS File System".

You can set disk quotas to restrict the amount of disk space (*blocks*) that users or groups can use, to limit the number of files (*inodes*) that users or groups can create, and to notify you when usage is reaching a

specified limit. A hard limit specifies the maximum number of blocks or inodes available to a user or group on the file system. Users or groups can exceed a soft limit for a period of time known as a *grace period*.

20.9.1 Enabling Disk Quotas on File Systems

To enable user or group disk quotas on a file system:

1. Install or update the quota package:

```
# yum install quota
```

2. Include the `usrquota` or `grpquota` options in the file system's `/etc/fstab` entry, for example:

```
/dev/sdb1        /home          ext4     usrquota,grpquota   0 0
```

3. Remount the file system:

```
# mount -o remount /home
```

4. Create the quota database files:

```
# quotacheck -cug /home
```

This command creates the files `aquota.user` and `aquota.group` in the root of the file system (`/home` in this example).

For more information, see the `quotacheck(8)` manual page.

20.9.2 Assigning Disk Quotas to Users and Groups

To configure the disk quota for a user:

1. Enter the following command for a user:

```
# edquota username
```

or for a group:

```
# edquota -g group
```

The command opens a text file opens in the default editor defined by the `EDITOR` environment variable, allowing you to specify the limits for the user or group, for example:

```
Disk quotas for user guest (uid 501)
Filesystem   blocks   soft   hard   inodes   soft   hard
 /dev/sdb1    10325      0      0     1054      0      0
```

The `blocks` and `inodes` entries show the user's currently usage on a file system.

> **Tip**
>
> Setting a limit to 0 disables quota checking and enforcement for the corresponding `blocks` or `inodes` category.

2. Edit the soft and hard block limits for number of blocks and inodes, and save and close the file.

Alternatively, you can use the `setquota` command to configure quota limits from the command-line. The `-p` option allows you to apply quota settings from one user or group to another user or group.

For more information, see the `edquota(8)` and `setquota(8)` manual pages.

20.9.3 Setting the Grace Period

To configure the grace period for soft limits:

1. Enter the following command:

    ```
    # edquota -t
    ```

    The command opens a text file opens in the default editor defined by the `EDITOR` environment variable, allowing you to specify the grace period, for example:

    ```
    Grace period before enforcing soft limits for users:
    Time units may be: days, hours, minutes, or seconds
      Filesystem       Block grace period      Inode grace period
      /dev/sdb1               7days                    7days
    ```

2. Edit the grace periods for the soft limits on the number of blocks and inodes, and save and close the file.

For more information, see the `edquota(8)` manual page.

20.9.4 Displaying Disk Quotas

To display a user's disk usage:

```
# quota username
```

To display a group's disk usage:

```
# quota -g group
```

To display information about file systems where usage is over the quota limits:

```
# quota -q
```

Users can also use the `quota` command to display their own and their group's usage.

For more information, see the `quota(1)` manual page.

20.9.5 Enabling and Disabling Disk Quotas

To disable disk quotas for all users, groups on a specific file system:

```
# quotaoff -guv filesystem
```

To disable disk quotas for all users, groups, and file systems:

```
# quotaoff -aguv
```

To re-enable disk quotas for all users, groups, and file systems:

```
# quotaon -aguv
```

For more information, see the `quotaon(1)` manual page.

20.9.6 Reporting on Disk Quota Usage

To display the disk quota usage for a file system:

```
# repquota filesystem
```

To display the disk quota usage for all file systems:

```
# repquota -a
```

For more information, see the `repquota(8)` manual page.

20.9.7 Maintaining the Accuracy of Disk Quota Reporting

Uncontrolled system shutdowns can lead to inaccuracies in disk quota reports.

To rebuild the quota database for a file system:

1. Disable disk quotas for the file system:

    ```
    # quotaoff -guv filesystem
    ```

2. Unmount the file system:

    ```
    # umount filesystem
    ```

3. Enter the following command to rebuild the quota databases:

    ```
    # quotacheck -guv filesystem
    ```

4. Mount the file system:

    ```
    # mount filesystem
    ```

5. Enable disk quotas for the file system:

    ```
    # quotaoff -guv filesystem
    ```

For more information, see the `quotacheck(8)` manual page.

Chapter 21 Local File System Administration

Table of Contents

This chapter describes administration tasks for the btrfs, ext3, ext4, OCFS2, and XFS local file systems.

21.1 About Local File Systems

Oracle Linux supports a large number of local file system types that you can configure on block devices, including:

`btrfs`	Btrfs is a copy-on-write file system that is designed to address the expanding scalability requirements of large storage subsystems. It supports snapshots, a roll-back capability, checksum functionality for

data integrity, transparent compression, and integrated logical volume management.

The maximum supported file or file system size is 50 TB.

For more information, see Section 21.2, "About the Btrfs File System".

ext3
The ext3 file system includes journaling capabilities to improve reliability and availability. Consistency checks after a power failure or an uncontrolled system shutdown are unnecessary. `ext2` file systems are upgradeable to ext3 without reformatting.

See Section 21.14, "Converting a Non-root Ext2 File System to Ext3" and Section 21.15, "Converting a root Ext2 File System to Ext3".

The maximum supported file and file system sizes are 2 TB and 16 TB.

ext4
In addition to the features of ext3, the ext4 file system supports extents (contiguous physical blocks), pre-allocation, delayed allocation, faster file system checking, more robust journaling, and other enhancements.

The maximum supported file or file system size is 50 TB.

ocfs2
Although intended as a general-purpose, high-performance, high-availability, shared-disk file system intended for use in clusters, it is possible to use Oracle Cluster File System version 2 (OCFS2) as a standalone, non-clustered file system.

Although it might seem that there is no benefit in mounting OCFS2 locally as compared to alternative file systems such as ext4 or btrfs, you can use the `reflink` command with OCFS2 to create copy-on-write clones of individual files in a similar way to using the `cp --reflink` command with the btrfs file system. Typically, such clones allow you to save disk space when storing multiple copies of very similar files, such as VM images or Linux Containers. In addition, mounting a local OCFS2 file system allows you to subsequently migrate it to a cluster file system without requiring any conversion.

See Section 21.16, "Creating a Local OCFS2 File System".

The maximum supported file or file system size is 16 TB.

vfat
The vfat file system (also known as FAT32) was originally developed for MS-DOS. It does not support journaling and lacks many of the features that are available with other file system types. It is mainly used to exchange data between Microsoft Windows and Oracle Linux systems.

The maximum supported file size or file system size is 2 GB.

xfs
XFS is a high-performance journaling file system, which provides high scalability for I/O threads, file system bandwidth, file and file system size, even when the file system spans many storage devices.

The maximum supported file and file system sizes are 16 TB and 500 TB respectively.

For more information, see Section 21.17, "About the XFS File System".

To see what file system types your system supports, use the following command:

```
# ls /sbin/mkfs.*
/sbin/mkfs.btrfs    /sbin/mkfs.ext3     /sbin/mkfs.msdos
/sbin/mkfs.cramfs   /sbin/mkfs.ext4     /sbin/mkfs.vfat
/sbin/mkfs.ext2     /sbin/mkfs.ext4dev  /sbin/mkfs.xfs
```

These executables are used to make the file system type specified by their extension. `mkfs.msdos` and `mkfs.vfat` are alternate names for `mkdosfs`. `mkfs.cramfs` creates a compressed ROM, read-only cramfs file system for use by embedded or small-footprint systems.

21.2 About the Btrfs File System

The btrfs file system is designed to meet the expanding scalability requirements of large storage subsystems. As the btrfs file system uses B-trees in its implementation, its name derives from the name of those data structures, although it is not a true acronym. A B-tree is a tree-like data structure that enables file systems and databases to efficiently access and update large blocks of data no matter how large the tree grows.

The btrfs file system provides the following important features:

- Copy-on-write functionality allows you to create both readable and writable snapshots, and to roll back a file system to a previous state, even after you have converted it from an `ext3` or `ext4` file system.

- Checksum functionality ensures data integrity.

- Transparent compression saves disk space.

- Transparent defragmentation improves performance.

- Integrated logical volume management allows you to implement RAID 0, RAID 1, or RAID 10 configurations, and to dynamically add and remove storage capacity.

> **Note**
>
> Configuring a swap file on a btrfs file system is not supported.

You can find more information about the btrfs file system at https://btrfs.wiki.kernel.org/index.php/Main_Page.

21.3 Creating a Btrfs File System

> **Note**
>
> If the `btrfs-progs` package is not already installed on your system, use `yum` to install it.

You can use the `mkfs.btrfs` command to create a btrfs file system that is laid out across one or more block devices. The default configuration is to stripe the file system data and to mirror the file system metadata across the devices. If you specify a single device, the metadata is duplicated on that device unless you specify that only one copy of the metadata is to be used. The devices can be simple disk partitions, loopback devices (that is, disk images in memory), multipath devices, or LUNs that implement RAID in hardware.

The following table illustrates how to use the `mkfs.btrfs` command to create various btrfs configurations.

Command	Description
`mkfs.btrfs` *`block_device`*	Create a btrfs file system on a single device. For example: `mkfs.btrfs /dev/sdb1`
`mkfs.btrfs -L` *`label`* *`block_device`*	Create a btrfs file system with a label that you can use when mounting the file system. For example: `mkfs.btrfs -L myvolume /dev/sdb2` **Note** The device must correspond to a partition if you intend to mount it by specifying the name of its label.
`mkfs.btrfs -m single` *`block_device`*	Create a btrfs file system on a single device, but do not duplicate the metadata on that device. For example: `mkfs.btrfs -m single /dev/sdc`
`mkfs.btrfs` *`block_device1`* *`block_device2`* ...	Stripe the file system data and mirror the file system metadata across several devices. For example: `mkfs.btrfs /dev/sdd /dev/sde`
`mkfs.btrfs -m raid0` *`block_device1`* *`block_device2`* ...	Stripe both the file system data and metadata across several devices. For example: `mkfs.btrfs -m raid0 /dev/sdd /dev/sde`
`mkfs.btrfs -d raid1` *`block_device1`* *`block_device2`* ...	Mirror both the file system data and metadata across several devices. For example: `mkfs.btrfs -d raid1 /dev/sdd /dev/sde`

`mkfs.btrfs -d raid10 -m raid10` *`block_device1`* *`block_device2`* *`block_device3`* *`block_device4`*	Stripe the file system data and metadata across several mirrored devices. You must specify an even number of devices, of which there must be at least four. For example:
	
	`mkfs.btrfs -d raid10 -m raid10 /dev/sdf \`
	
	`/dev/sdg /dev/sdh /dev/sdi /dev/sdj /dev/`
	`sdk`

When you want to mount the file system, you can specify it by any of its component devices, for example:

```
# mkfs.btrfs -d raid10 -m raid10 /dev/sd[fghijk]
# mount /dev/sdf /raid10_mountpoint
```

To find out the RAID configuration of a mounted btrfs file system, use this command:

```
# btrfs filesystem df mountpoint
```

Note

The `btrfs filesystem df` command displays more accurate information about the space used by a btrfs file system than the `df` command does.

Use the following form of the `btrfs` command to display information about all the btrfs file systems on a system:

```
# btrfs filesystem show
```

21.4 Modifying a Btrfs File System

The following table shows how you can use the `btrfs` command to add or remove devices, and to rebalance the layout of the file system data and metadata across the devices.

Command	Description
`btrfs device add` *device mountpoint*	Add a device to the file system that is mounted on the specified mount point. For example: `btrfs device add /dev/sdd /myfs`
`btrfs device delete` *device mountpoint*	Remove a device from a mounted file system. For example: `btrfs device delete /dev/sde /myfs`
`btrfs device delete missing` *mountpoint*	Remove a failed device from the file system that is mounted in degraded mode. For example: `btrfs device remove missing /myfs` To mount a file system in degraded mode, specify the `-o degraded` option to the `mount` command. For a RAID configuration, if the number of devices would fall below the minimum number that are required, you must add the replacement device before removing the failed device.
`btrfs filesystem balance` *mountpoint*	After adding or removing devices, redistribute the file system data and metadata across the available devices.

21.5 Compressing and Defragmenting a Btrfs File System

You can compress a btrfs file system to increase its effective capacity, and you can defragment it to increase I/O performance.

To enable compression of a btrfs file system, specify one of the following `mount` options:

Mount Option	Description
`compress=lzo`	Use LZO compression.
`compress=zlib`	Use zlib compression.

LZO offers a better compression ratio, while zlib offers faster compression.

You can also compress a btrfs file system at the same time that you defragment it.

To defragment a btrfs file system, use the following command:

```
# btrfs filesystem defragment filesystem_name
```

To defragment a btrfs file system and compress it at the same time:

```
# btrfs filesystem defragment -c filesystem_name
```

You can also defragment, and optionally compress, individual file system objects, such as directories and files, within a btrfs file system.

```
# btrfs filesystem defragment [-c] file_name ...
```

> **Note**
>
> You can set up automatic defragmentation by specifying the `autodefrag` option when you mount the file system. However, automatic defragmentation is not recommended for large databases or for images of virtual machines.
>
> Defragmenting a file or a subvolume that has a copy-on-write copy results breaks the link between the file and its copy. For example, if you defragment a subvolume that has a snapshot, the disk usage by the subvolume and its snapshot will increase because the snapshot is no longer a copy-on-write image of the subvolume.

21.6 Resizing a Btrfs File System

You can use the `btrfs` command to increase the size of a mounted btrfs file system if there is space on the underlying devices to accommodate the change, or to decrease its size if the file system has sufficient available free space. The command does not have any effect on the layout or size of the underlying devices.

For example, to increase the size of `/mybtrfs1` by 2 GB:

```
# btrfs filesystem resize +2g /mybtrfs1
```

Decrease the size of `/mybtrfs2` by 4 GB:

```
# btrfs filesystem resize -4g /mybtrfs2
```

Set the size of `/mybtrfs3` to 20 GB:

```
# btrfs filesystem resize 20g /mybtrfs3
```

21.7 Creating Subvolumes and Snapshots

The top level of a btrfs file system is a subvolume consisting of a named b-tree structure that contains directories, files, and possibly further btrfs subvolumes that are themselves named b-trees that contain directories and files, and so on. To create a subvolume, change directory to the position in the btrfs file system where you want to create the subvolume and enter the following command:

```
# btrfs subvolume create subvolume_name
```

Snapshots are a type of subvolume that records the contents of their parent subvolumes at the time that you took the snapshot. If you take a snapshot of a btrfs file system and do not write to it, the snapshot records the state of the original file system and forms a stable image from which you can make a backup. If you make a snapshot writable, you can treat it as a alternate version of the original file system. The copy-on-write functionality of btrfs file system means that snapshots are quick to create, and consume very little disk space initially.

> **Note**
>
> Taking snapshots of a subvolume is not a recursive process. If you create a snapshot of a subvolume, every subvolume or snapshot that the subvolume contains is mapped to an empty directory of the same name inside the snapshot.

The following table shows how to perform some common snapshot operations:

Command	Description
`btrfs subvolume snapshot` *`pathname`* *`pathname`*`/`*`snapshot_path`*	Create a snapshot *`snapshot_path`* of a parent subvolume or snapshot specified by *`pathname`*. For example: `btrfs subvolume snapshot /mybtrfs /mybtrfs/snapshot1`
`btrfs subvolume list` *`pathname`*	List the subvolumes or snapshots of a subvolume or snapshot specified by *`pathname`*. For example: `btrfs subvolume list /mybtrfs` **Note** You can use this command to determine the ID of a subvolume or snapshot.
`btrfs subvolume set-default` *`ID`* *`pathname`*	By default, mount the snapshot or subvolume specified by its ID instead of the parent subvolume. For example: `btrfs subvolume set-default 4 /mybtrfs`
`btrfs subvolume get-default` *`pathname`*	Displays the ID of the default subvolume that is mounted for the specified subvolume. For example: `btrfs subvolume get-default /mybtrfs`

You can mount a btrfs subvolume as though it were a disk device. If you mount a snapshot instead of its parent subvolume, you effectively roll back the state of the file system to the time that the snapshot was taken. By default, the operating system mounts the parent btrfs volume, which has an ID of 0, unless you use `set-default` to change the default subvolume. If you set a new default subvolume, the system will mount that subvolume instead in future. You can override the default setting by specifying either of the following `mount` options:

Mount Option	Description
`subvolid=`*`snapshot-ID`*	Mount the subvolume or snapshot specified by its subvolume ID instead of the default subvolume.
`subvol=`*`pathname`*`/`*`snapshot_path`*	Mount the subvolume or snapshot specified by its pathname instead of the default subvolume. **Note** The subvolume or snapshot must be located in the root of the btrfs file system.

When you have rolled back a file system by mounting a snapshot, you can take snapshots of the snapshot itself to record its state.

When you no longer require a subvolume or snapshot, use the following command to delete it:

```
# btrfs subvolume delete subvolume_path
```

> **Note**
>
> Deleting a subvolume deletes all subvolumes that are below it in the b-tree hierarchy. For this reason, you cannot remove the topmost subvolume of a btrfs file system, which has an ID of 0.

For details of how to use the `snapper` command to create and manage btrfs snapshots, see Section 21.7.1, "Using snapper with Btrfs Subvolumes".

21.7.1 Using snapper with Btrfs Subvolumes

You can use the `snapper` utility to create and manage snapshots of btrfs subvolumes.

To set up the snapper configuration for an existing mounted btrfs subvolume:

```
# snapper -c config_name create-config -f btrfs fs_name
```

Here `config_name` is the name of the configuration and `fs_name` is the path of the mounted btrfs subvolume. The command adds an entry for `config_name` to `/etc/sysconfig/snapper`, creates the configuration file `/etc/snapper/configs/config_name`, and sets up a `.snapshots` subvolume for the snapshots.

For example, the following command sets up the `snapper` configuration for a btrfs root file system:

```
# snapper -c root create-config -f btrfs /
```

By default, `snapper` sets up a `cron.hourly` job to create snapshots in the `.snapshot` subdirectory of the subvolume and a `cron.daily` job to clean up old snapshots. You can edit the configuration file to disable or change this behavior. For more information, see the `snapper-configs(5)` manual page.

There are three types of snapshot that you can create using `snapper`:

`post`	You use a *post snapshot* to record the state of a subvolume after a modification. A post snapshot should always be paired with a *pre snapshot* that you take immediately before you make the modification.
`pre`	You use a *pre snapshot* to record the state of a subvolume before a modification. A pre snapshot should always be paired with a *post snapshot* that you take immediately after you have completed the modification.
`single`	You can use a single snapshot to record the state of a subvolume but it does not have any association with other snapshots of the subvolume.

For example, the following commands create pre and post snapshots of a subvolume:

```
# snapper -c config_name create -t pre -p
N
... Modify the subvolume's contents...
# snapper -c config_name create -t post --pre-num N -p
N'
```

The `-p` option causes `snapper` to display the number of the snapshot so that you can reference it when you create the post snapshot or when you compare the contents of the pre and post snapshots.

To display the files and directories that have been added, removed, or modified between the pre and post snapshots, use the `status` subcommand:

```
# snapper -c config_name status N..N'
```

To display the differences between the contents of the files in the pre and post snapshots, use the `diff` subcommand:

```
# snapper -c config_name diff N..N'
```

To list the snapshots that exist for a subvolume:

```
# snapper -c config_name list
```

To delete a snapshot, specify its number to the `delete` subcommand:

```
# snapper -c config_name delete N''
```

To undo the changes in the subvolume from post snapshot `N'` to pre snapshot `N`:

```
# snapper -c config_name undochange N..N'
```

For more information, see the `snapper(8)` manual page.

21.7.2 Cloning Virtual Machine Images and Linux Containers

You can use a btrfs file system to provide storage space for virtual machine images and Linux Containers. The ability to quickly clone files and create snapshots of directory structures makes btrfs an ideal candidate for this purpose. For details of how to use the snapshot feature of btrfs to implement Linux Containers, see Chapter 28, *Linux Containers*.

21.8 Using the Send/Receive Feature

Note

The send/receive feature requires that you boot the system using UEK R3.

The send operation compares two subvolumes and writes a description of how to convert one subvolume (the *parent* subvolume) into the other (the *sent* subvolume). You would usually direct the output to a file for later use or pipe it to a receive operation for immediate use.

The simplest form of the send operation writes a complete description of a subvolume:

```
# btrfs send [-v] [-f sent_file] ... subvol
```

You can specify multiple instances of the `-v` option to display increasing amounts of debugging output. The `-f` option allows you to save the output to a file. Both of these options are implicit in the following usage examples.

The following form of the send operation writes a complete description of how to convert one subvolume into another:

```
# btrfs send -p parent_subvol sent_subvol
```

If a subvolume such as a snapshot of the parent volume, known as a *clone source*, will be available during the receive operation from which some of the data can be recovered, you can specify the clone source to reduce the size of the output file:

```
# btrfs send [-p parent_subvol] -c clone_src [-c clone_src] ... subvol
```

You can specify the `-c` option multiple times if there is more than one clone source. If you do not specify the parent subvolume, btrfs chooses a suitable parent from the clone sources.

You use the receive operation to regenerate the sent subvolume at a specified path:

```
# btrfs receive [-f sent_file] mountpoint
```

21.8.1 Using Send/Receive to Implement Incremental Backups

The following procedure is a suggestion for setting up an incremental backup and restore process for a subvolume.

1. Create a read-only snapshot of the subvolume to serve as an initial reference point for the backup:

   ```
   # btrfs subvolume snapshot -r /vol /vol/backup_0
   ```

2. Run `sync` to ensure that the snapshot has been written to disk:

   ```
   # sync
   ```

3. Create a subvolume or directory on a btrfs file system as a backup area to receive the snapshot, for example, `/backupvol`.

4. Send the snapshot to `/backupvol`:

   ```
   # btrfs send /vol/backup_0 | btrfs receive /backupvol
   ```

   This command creates the subvolume `/backupvol/backup_0`.

   Having created the reference backup, you can then create incremental backups as required.

5. To create an incremental backup:

   a. Create a new snapshot of the subvolume:

      ```
      # btrfs subvolume snapshot -r /vol /vol/backup_1
      ```

   b. Run `sync` to ensure that the snapshot has been written to disk:

      ```
      # sync
      ```

   c. Send only the differences between the reference backup and the new backup to the backup area:

      ```
      # btrfs send -p /vol/backup_0 /vol/backup_1 | btrfs receive /backupvol
      ```

      This command creates the subvolume `/backupvol/backup_1`.

21.9 Using Quota Groups

> **Note**
>
> The quota groups feature requires that you boot the system using UEK R3.

To enable quotas, use the following command on a newly created btrfs file system before any creating any subvolumes:

```
# btrfs quota enable volume
```

To assign a quota-group limit to a subvolume, use the following command:

```
# btrfs qgroup limit size /volume/subvolume
```

For example:

```
# btrfs qgroup limit 1g /myvol/subvol1
# btrfs qgroup limit 512m /myvol/subvol2
```

To find out the quota usage for a subvolume, use the `btrfs qgroup show path` command:

21.10 Replacing Devices on a Live File System

Note

The device replacement feature requires that you boot the system using UEK R3.

You can replace devices on a live file system. You do not need to unmount the file system or stop any tasks that are using it. If the system crashes or loses power while the replacement is taking place, the operation resumes when the system next mounts the file system.

Use the following command to replace a device on a mounted btrfs file system:

```
# btrfs replace start source_dev target_dev [-r] mountpoint
```

`source_dev` and `target_dev` specify the device to be replaced (*source device*) and the replacement device (*target device*). `mountpoint` specifies the file system that is using the source device. The target device must be the same size as or larger than the source device. If the source device is no longer available or you specify the `-r` option, the data is reconstructed by using redundant data obtained from other devices (such as another available mirror). The source device is removed from the file system when the operation is complete.

You can use the `btrfs replace status mountpoint` and `btrfs replace cancel mountpoint` commands to check the progress of the replacement operation or to cancel the operation.

21.11 Creating Snapshots of Files

You can use the `--reflink` option to the `cp` command to create lightweight copies of a file within the same subvolume of a btrfs file system. The copy-on-write mechanism saves disk space and allows copy operations to be almost instantaneous. The btrfs file system creates a new inode that shares the same disk blocks as the existing file, rather than creating a complete copy of the file's data or creating a link that points to the file's inode. The resulting file appears to be a copy of the original file, but the original data blocks are not duplicated. If you subsequently write to one of the files, the btrfs file system makes copies of the blocks before they are written to, preserving the other file's content.

For example, the following command creates the snapshot `bar` of the file `foo`:

```
# cp -reflink foo bar
```

21.12 Converting an Ext2, Ext3, or Ext4 File System to a Btrfs File System

You can use the `btrfs-convert` utility to convert an `ext2`, `ext3`, or `ext4` file system to `btrfs`. The utility preserves an image of the original file system in a snapshot named `ext2_saved`. This snapshot allows you to roll back the conversion, even if you have made changes to the btrfs file system.

Note

You cannot convert the root file system or a bootable partition, such as `/boot`, to btrfs.

21.12.1 Converting a Non-root File System

Caution

Before performing a file system conversion, make a backup of the file system from which you can restore its state.

To convert an `ext2`, `ext3`, or `ext4` file system other than the root file system to `btrfs`:

1. Unmount the file system.

   ```
   # umount mountpoint
   ```

2. Run the correct version of `fsck` (for example, `fsck.ext4`) on the underlying device to check and correct the integrity of file system.

   ```
   # fsck.extN -f device
   ```

3. Convert the file system to a btrfs file system.

   ```
   # btrfs-convert device
   ```

4. Edit the file `/etc/fstab`, and change the file system type of the file system to `btrfs`, for example:

   ```
   /dev/sdb                /myfs           btrfs   defaults  0 0
   ```

5. Mount the converted file system on the old mount point.

   ```
   # mount device mountpoint
   ```

21.13 About the Btrfs root File System

Oracle Linux 7 installation allows you to create a btrfs root file system. The mounted root file system is a snapshot (named `install`) of the root file system taken at the end of installation. To find out the ID of the parent of the root file system subvolume, use the following command:

```
# btrfs subvolume list /
ID 258 top level 5 path install
```

In this example, the installation root file system subvolume has an ID of 5. The subvolume with ID 258 (`install`) is currently mounted as `/`. Figure 21.1, "Layout of the root File System Following Installation" illustrates the layout of the file system:

Figure 21.1 Layout of the root File System Following Installation
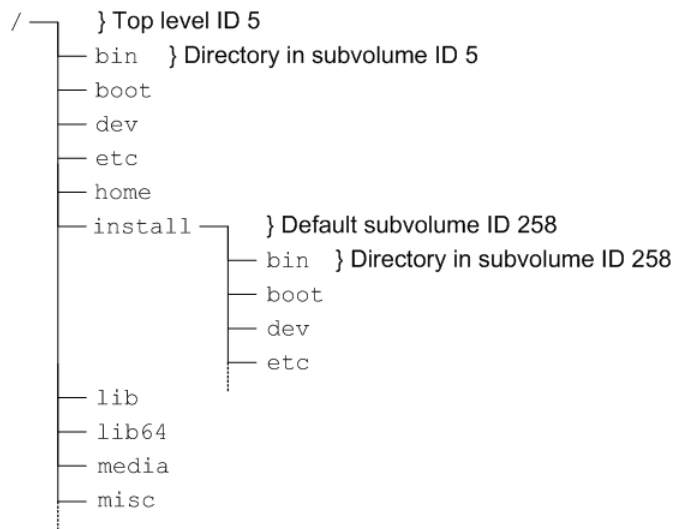
The top-level subvolume with ID 5 records the contents of the root file system file system at the end of installation. The default subvolume (`install`) with ID 258 is currently mounted as the active root file system.

The `mount` command shows the device that is currently mounted as the root file system:

```
# mount
/dev/mapper/vg_btrfs-lv_root on / type btrfs (rw)
...
```

To mount the installation root file system volume, you can use the following commands:

```
# mkdir /instroot
# mount -o subvolid=5 /dev/mapper/vg-btrfs-lv-root /instroot
```

If you list the contents of `/instroot`, you can see both the contents of the installation root file system volume and the `install` snapshot, for example:

```
# ls /instroot
bin   cgroup  etc    install  lib64  misc  net  proc  sbin     srv  tmp  var
boot  dev     home   lib      media  mnt   opt  root  selinux  sys  usr
```

The contents of `/` and `/instroot/install` are identical as demonstrated in the following example where a file (`foo`) created in `/instroot/install` is also visible in `/`:

```
# touch /instroot/install/foo
# ls /
bin   cgroup  etc  home      lib    media  mnt  opt   root  selinux  sys  usr
boot  dev     foo  instroot  lib64  misc   net  proc  sbin  srv      tmp  var
# ls /instroot/install
bin   cgroup  etc  home      lib    media  mnt  opt   root  selinux  sys  usr
boot  dev     foo  instroot  lib64  misc   net  proc  sbin  srv      tmp  var
# rm -f /foo
# ls /
bin   cgroup  etc    instroot  lib64  misc  net  proc  sbin     srv  tmp  var
boot  dev     home   lib       media  mnt   opt  root  selinux  sys  usr
# ls /instroot/install
bin   cgroup  etc    instroot  lib64  misc  net  proc  sbin     srv  tmp  var
boot  dev     home   lib       media  mnt   opt  root  selinux  sys  usr
```

21.13.1 Creating Snapshots of the root File System

To take a snapshot of the current root file system:

1. Mount the top level of the root file system on a suitable mount point.

   ```
   # mount -o subvolid=5 /dev/mapper/vg-btrfs-lv-root /mnt
   ```

2. Change directory to the mount point and take the snapshot. In this example, the `install` subvolume is currently mounted as the root file system system.

   ```
   # cd /mnt
   # btrfs subvolume snapshot install root_snapshot_1
   Create a snapshot of 'install' in './root_snapshot_1'
   ```

3. Change directory to `/` and unmount the top level of the file system.

   ```
   # cd /
   # umount /mnt
   ```

   The list of subvolumes now includes the newly created snapshot.

   ```
   # btrfs subvolume list /
   ID 258 top level 5 path install
   ID 260 top level 5 path root_snapshot_1
   ```

21.13.2 Mounting Alternate Snapshots as the root File System

If you want to roll back changes to your system, you can mount a snapshot as the root file system by specifying its ID as the default subvolume, for example:

```
# btrfs subvolume set-default 260 /
```

Reboot the system for the change to take effect.

21.13.3 Deleting Snapshots of the root File System

To delete a snapshot:

1. Mount the top level of the file system, for example:

    ```
    # mount -o subvolid=5 /dev/mapper/vg-btrfs-lv-root /mnt
    ```

2. Change directory to the mount point and delete the snapshot.

    ```
    # cd /mnt
    # btrfs subvolume delete install
    Delete subvolume '/mnt/install'
    ```

3. Change directory to `/` and unmount the top level of the file system.

    ```
    # cd /
    # umount /mnt
    ```

    The list of subvolumes now does not include `install`.

    ```
    # btrfs subvolume list /
    ID 260 top level 5 path root_snapshot_1
    ```

21.14 Converting a Non-root Ext2 File System to Ext3

Caution

Before performing a file system conversion, make a backup of the file system from which you can restore its state.

To convert a non-root ext2 file system to ext3:

1. Unmount the ext2 file system:

    ```
    # umount filesystem
    ```

2. Use `fsck.ext2` to check the file system.

    ```
    bash-4.1# fsck.ext2 -f device
    ```

3. Use the following command with the block device corresponding to the ext2 file system:

    ```
    # tune2fs -j device
    ```

    The command adds an ext3 journal inode to the file system.

4. Use `fsck.ext3` to check the file system.

    ```
    bash-4.1# fsck.ext3 -f device
    ```

5. Correct any entry for the file system in `/etc/fstab` so that its type is defined as `ext3` instead of `ext2`.

6. You can now remount the file system whenever convenient:

```
# mount filesystem
```

For more information, see the `tune2fs(8)` manual page.

21.15 Converting a root Ext2 File System to Ext3

⚠️ **Caution**

Before performing a root file system conversion, make a full system backup from which you can restore its state.

To convert a root ext2 file system to ext3:

1. Use the following command with the block device corresponding to the root file system:

```
# tune2fs -j device
```

The command adds an ext3 journal to the file system as the file `/.journal`.

2. Run the `mount` command to determine the device that is currently mounted as the root file system.

In the following example, the root file system corresponds to the disk partition `/dev/sda2`:

```
# mount
/dev/sda2 on / type ext2 (rw)
```

3. Shut down the system.

4. Boot the system from an Oracle Linux boot CD, DVD or ISO. You can download the ISO from https://edelivery.oracle.com/linux.

5. From the installation menu, select **Rescue Installed System**. When prompted, choose a language and keyboard, select **Local CD/DVD** as the installation media, select **No** to bypass starting the network interface, and select **Skip** to bypass selecting a rescue environment.

6. Select **Start shell** to obtain a `bash` shell prompt (`bash-4.1#`) at the bottom of the screen.

7. If the existing root file system is configured as an LVM volume, use the following command to start the volume group (for example, `vg_host01`):

```
bash-4.1# lvchange -ay vg_host01
```

8. Use `fsck.ext3` to check the file system.

```
bash-4.1# fsck.ext3 -f device
```

where `device` is the root file system device (for example, `/dev/sda2`).

The command moves the `.journal` file to the journal inode.

9. Create a mount point (`/mnt1`) and mount the converted root file system on it.

```
bash-4.1# mkdir /mnt1
bash-4.1# mount -t ext3 device /mnt1
```

10. Use the `vi` command to edit `/mnt1/etc/fstab`, and change the file system type of the root file system to `ext3`, for example:

```
/dev/sda2        /        ext3      defaults  1 1
```

11. Create the file `.autorelabel` in the root of the mounted file system.

```
bash-4.1# touch /mnt1/.autorelabel
```

The presence of the `.autorelabel` file in `/` instructs SELinux to recreate the security attributes of all files on the file system.

> **Note**
>
> If you do not create the `.autorelabel` file, you might not be able to boot the system successfully. If you forget to create the file and the reboot fails, either disable SELinux temporarily by specifying `selinux=0` to the kernel boot parameters, or run SELinux in permissive mode by specifying `enforcing=0`.

12. Unmount the converted root file system.

```
bash-4.1# umount /mnt1
```

13. Remove the boot CD, DVD, or ISO, and reboot the system.

For more information, see the `tune2fs(8)` manual page.

21.16 Creating a Local OCFS2 File System

To create an OCFS2 file system that will be locally mounted and not associated with a cluster, use the following command:

```
# mkfs.ocfs2 -M local --fs-features=local -N 1 [options] device
```

For example, create a locally mountable OCFS2 volume on `/dev/sdc1` with one node slot and the label `localvol`:

```
# mkfs.ocfs2 -M local --fs-features=local -N 1 -L "localvol" /dev/sdc1
```

You can use the `tunefs.ocfs2` utility to convert a local OCTFS2 file system to cluster use, for example:

```
# umount /dev/sdc1
# tunefs.ocfs2 -M cluster --fs-features=cluster -N 8 /dev/sdc1
```

This example also increases the number of node slots from 1 to 8 to allow up to eight nodes to mount the file system.

For information about using OCFS2 with clusters, see Chapter 23, *Oracle Cluster File System Version 2*.

21.17 About the XFS File System

XFS is a high-performance journaling file system that was initially created by Silicon Graphics, Inc. for the IRIX operating system and later ported to Linux. The parallel I/O performance of XFS provides high scalability for I/O threads, file system bandwidth, file and file system size, even when the file system spans many storage devices.

A typical use case for XFS is to implement a several-hundred terabyte file system across multiple storage servers, each server consisting of multiple FC-connected disk arrays.

XFS is supported for use with the root (`/`) or `boot` file systems on Oracle Linux 7.

XFS has a large number of features that make it suitable for deployment in an enterprise-level computing environment that requires the implementation of very large file systems:

- XFS implements journaling for metadata operations, which guarantees the consistency of the file system following loss of power or a system crash. XFS records file system updates asynchronously to a circular buffer (the *journal*) before it can commit the actual data updates to disk. The journal can be located either internally in the data section of the file system, or externally on a separate device to reduce contention for disk access. If the system crashes or loses power, it reads the journal when the file system is remounted, and replays any pending metadata operations to ensure the consistency of the file system. The speed of this recovery does not depend on the size of the file system.

- XFS is internally partitioned into allocation groups, which are virtual storage regions of fixed size. Any files and directories that you create can span multiple allocation groups. Each allocation group manages its own set of inodes and free space independently of other allocation groups to provide both scalability and parallelism of I/O operations. If the file system spans many physical devices, allocation groups can optimize throughput by taking advantage of the underlying separation of channels to the storage components.

- XFS is an extent-based file system. To reduce file fragmentation and file scattering, each file's blocks can have variable length extents, where each extent consists of one or more contiguous blocks. XFS's space allocation scheme is designed to efficiently locate free extents that it can use for file system operations. XFS does not allocate storage to the holes in sparse files. If possible, the extent allocation map for a file is stored in its inode. Large allocation maps are stored in a data structure maintained by the allocation group.

- To maximize throughput for XFS file systems that you create on an underlying striped, software or hardware-based array, you can use the `su` and `sw` arguments to the `-d` option of the `mkfs.xfs` command to specify the size of each stripe unit and the number of units per stripe. XFS uses the information to align data, inodes, and journal appropriately for the storage. On `lvm` and `md` volumes and some hardware RAID configurations, XFS can automatically select the optimal stripe parameters for you.

- To reduce fragmentation and increase performance, XFS implements *delayed allocation*, reserving file system blocks for data in the buffer cache, and allocating the block when the operating system flushes that data to disk.

- XFS supports extended attributes for files, where the size of each attribute's value can be up to 64 KB, and each attribute can be allocated to either a `root` or a `user` name space.

- Direct I/O in XFS implements high throughput, non-cached I/O by performing DMA directly between an application and a storage device, utilising the full I/O bandwidth of the device.

- To support the snapshot facilities that volume managers, hardware subsystems, and databases provide, you can use the `xfs_freeze` command to suspend and resume I/O for an XFS file system. See Section 21.22, "Freezing and Unfreezing an XFS File System".

- To defragment individual files in an active XFS file system, you can use the `xfs-fsr` command. See Section 21.25, "Defragmenting an XFS File System".

- To grow an XFS file system, you can use the `xfs_growfs` command. See Section 21.21, "Growing an XFS File System".

- To back up and restore a live XFS file system, you can use the `xfsdump` and `xfsrestore` commands. See Section 21.24, "Backing up and Restoring XFS File Systems".

- XFS supports user, group, and project disk quotas on block and inode usage that are initialized when the file system is mounted. Project disk quotas allow you to set limits for individual directory hierarchies

within an XFS file system without regard to which user or group has write access to that directory hierarchy.

You can find more information about XFS at http://xfs.org/index.php/XFS_Papers_and_Documentation.

21.17.1 About External XFS Journals

The default location for an XFS journal is on the same block device as the data. As synchronous metadata writes to the journal must complete successfully before any associated data writes can start, such a layout can lead to disk contention for the typical workload pattern on a database server. To overcome this problem, you can place the journal on a separate physical device with a low-latency I/O path. As the journal typically requires very little storage space, such an arrangement can significantly improve the file system's I/O throughput. A suitable host device for the journal is a solid-state drive (SSD) device or a RAID device with a battery-backed write-back cache.

To reserve an external journal with a specified size when you create an XFS file system, specify the `-l logdev=device,size=size` option to the `mkfs.xfs` command. If you omit the `size` parameter, `mkfs.xfs` selects a journal size based on the size of the file system. To mount the XFS file system so that it uses the external journal, specify the `-o logdev=device` option to the `mount` command.

21.17.2 About XFS Write Barriers

A write barrier assures file system consistency on storage hardware that supports flushing of in-memory data to the underlying device. This ability is particularly important for write operations to an XFS journal that is held on a device with a volatile write-back cache.

By default, an XFS file system is mounted with a write barrier. If you create an XFS file system on a LUN that has a battery-backed, non-volatile cache, using a write barrier degrades I/O performance by requiring data to be flushed more often than necessary. In such cases, you can remove the write barrier by mounting the file system with the `-o nobarrier` option to the `mount` command.

21.17.3 About Lazy Counters

With lazy-counters enabled on an XFS file system, the free-space and inode counters are maintained in parts of the file system other than the superblock. This arrangement can significantly improve I/O performance for application workloads that are metadata intensive.

Lazy counters are enabled by default, but if required, you can disable them by specifying the `-l lazy-count=0` option to the `mkfs.xfs` command.

21.18 Installing the XFS Packages

Note

You can also obtain the XFS packages from the Oracle Linux Yum Server.

To install the XFS packages on a system:

1. Log in to ULN, and subscribe your system to the `ol7_x86_64_latest` channel.

2. On your system, use `yum` to install the `xfsprogs` and `xfsdump` packages:

   ```
   # yum install xfsprogs xfsdump
   ```

3. If you require the XFS development and QA packages, additionally subscribe your system to the `ol7_x86_64_optional` channel and use `yum` to install them:

```
# yum install xfsprogs-devel xfsprogs-qa-devel
```

21.19 Creating an XFS File System

You can use the `mkfs.xfs` command to create an XFS file system, for example.

```
# mkfs.xfs /dev/vg0/lv0
meta-data=/dev/vg0/lv0              isize=256   agcount=32, agsize=8473312 blks
         =                         sectsz=512  attr=2, projid32bit=0
data     =                         bsize=4096  blocks=271145984, imaxpct=25
         =                         sunit=0     swidth=0 blks
naming   =version 2               bsize=4096  ascii-ci=0
log      =internal log           bsize=4096  blocks=32768, version=2
         =                         sectsz=512  sunit=0 blks, lazy-count=1
realtime =none                    extsz=4096  blocks=0, rtextents=0
```

To create an XFS file system with a stripe-unit size of 32 KB and 6 units per stripe, you would specify the `su` and `sw` arguments to the `-d` option, for example:

```
# mkfs.xfs -d su=32k,sw=6 /dev/vg0/lv1
```

For more information, see the `mkfs.xfs(8)` manual page.

21.20 Modifying an XFS File System

Note

You cannot modify a mounted XFS file system.

You can use the `xfs_admin` command to modify an unmounted XFS file system. For example, you can enable or disable lazy counters, change the file system UUID, or change the file system label.

To display the existing label for an unmounted XFS file system and then apply a new label:

```
# xfs_admin -l /dev/sdb
label = ""
# xfs_admin -L "VideoRecords" /dev/sdb
writing all SBs
new label = "VideoRecords"
```

Note

The label can be a maximum of 12 characters in length.

To display the existing UUID and then generate a new UUID:

```
# xfs_admin -u /dev/sdb
UUID = cd4f1cc4-15d8-45f7-afa4-2ae87d1db2ed
# xfs_admin -U generate /dev/sdb
writing all SBs
new UUID = c1b9d5a2-f162-11cf-9ece-0020afc76f16
```

To clear the UUID altogether:

```
# xfs_admin -U nil /dev/sdb
Clearing log and setting UUID
writing all SBs
new UUID = 00000000-0000-0000-0000-000000000000
```

To disable and then re-enable lazy counters:

```
# xfs_admin -c 0 /dev/sdb
Disabling lazy-counters
# xfs_admin -c 1 /dev/sdb
Enabling lazy-counters
```

For more information, see the mkfs_admin(8) manual page.

21.21 Growing an XFS File System

> **Note**
>
> You cannot grow an XFS file system that is currently unmounted.
>
> There is currently no command to shrink an XFS file system.

You can use the xfs_growfs command to increase the size of a mounted XFS file system if there is space on the underlying devices to accommodate the change. The command does not have any effect on the layout or size of the underlying devices. If necessary, use the underlying volume manager to increase the physical storage that is available. For example, you can use the vgextend command to increase the storage that is available to an LVM volume group and lvextend to increase the size of the logical volume that contains the file system.

You cannot use the parted command to resize a partition that contains an XFS file system. You must instead recreate the partition with a larger size and restore its contents from a backup if you deleted the original partition or from the contents of the original partition if you did not delete it to free up disk space.

For example, to increase the size of /myxfs1 to 4 TB, assuming a block size of 4 KB:

```
# xfs_growfs -D 1073741824 /myxfs1
```

To increase the size of the file system to the maximum size that the underlying device supports, specify the -d option:

```
# xfs_growfs -d /myxfs1
```

For more information, see the xfs_growfs(8) manual page.

21.22 Freezing and Unfreezing an XFS File System

If you need to take a hardware-based snapshot of an XFS file system, you can temporarily stop write operations to it.

> **Note**
>
> You do not need to explicitly suspend write operations if you use the lvcreate command to take an LVM snapshot.

To freeze and unfreeze an XFS file system, use the -f and -u options with the xfs_freeze command, for example:

```
# xfs_freeze -f /myxfs
# # ... Take snapshot of file system ...
# xfs_freeze -u /myxfs
```

> **Note**
>
> You can also use the xfs_freeze command with btrfs, ext3, and ext4 file systems.

For more information, see the `xfs_freeze(8)` manual page.

21.23 Setting Quotas on an XFS File System

The following table shows the `mount` options that you can specify to enable quotas on an XFS file system:

Mount Option	Description
gqnoenforce	Enable group quotas. Report usage, but do not enforce usage limits.
gquota	Enable group quotas and enforce usage limits.
pqnoenforce	Enable project quotas. Report usage, but do not enforce usage limits.
pquota	Enable project quotas and enforce usage limits.
uqnoenforce	Enable user quotas. Report usage, but do not enforce usage limits.
uquota	Enable user quotas and enforce usage limits.

To show the block usage limits and the current usage in the `myxfs` file system for all users, use the `xfs_quota` command:

```
# xfs_quota -x -c 'report -h' /myxfs
User quota on /myxfs (/dev/vg0/lv0)
                        Blocks
User ID      Used    Soft    Hard Warn/Grace
---------- -------------------------------
root              0      0      0  00 [------]
guest             0   200M   250M  00 [------]
```

The following forms of the command display the free and used counts for blocks and inodes respectively in the manner of the `df -h` command:

```
# xfs_quota -c 'df -h' /myxfs
Filesystem      Size   Used  Avail Use% Pathname
/dev/vg0/lv0 200.0G  32.2M  20.0G   1% /myxfs

# xfs_quota -c 'df -ih' /myxfs
Filesystem   Inodes   Used   Free Use% Pathname
/dev/vg0/lv0  21.0m      4  21.0m   1% /myxfs
```

If you specify the `-x` option to enter expert mode, you can use subcommands such as `limit` to set soft and hard limits for block and inode usage by an individual user, for example:

```
# xfs_quota -x -c 'limit bsoft=200m bhard=250m isoft=200 ihard=250 guest' /myxfs
```

Of course, this command requires that you mounted the file system with user quotas enabled.

To set limits for a group on an XFS file system that you have mounted with group quotas enabled, specify the `-g` option to `limit`, for example:

```
# xfs_quota -x -c 'limit -g bsoft=5g bhard=6g devgrp' /myxfs
```

For more information, see the `xfs_quota(8)` manual page.

21.23.1 Setting Project Quotas

User and group quotas are supported by other file systems, such as `ext4`. The XFS file system additionally allows you to set quotas on individual directory hierarchies in the file system that are known as *managed trees*. Each managed tree is uniquely identified by a *project ID* and an optional *project name*. Being able to control the disk usage of a directory hierarchy is useful if you do not otherwise want to set

quota limits for a privileged user (for example, `/var/log`) or if many users or groups have write access to a directory (for example, `/var/tmp`).

To define a project and set quota limits on it:

1. Mount the XFS file system with project quotas enabled:

    ```
    # mount -o pquota device mountpoint
    ```

    For example, to enable project quotas for the `/myxfs` file system:

    ```
    # mount -o pquota /dev/vg0/lv0 /myxfs
    ```

2. Define a unique project ID for the directory hierarchy in the `/etc/projects` file:

    ```
    # echo project_ID:mountpoint/directory >> /etc/projects
    ```

    For example, to set a project ID of 51 for the directory hierarchy `/myxfs/testdir`:

    ```
    # echo 51:/myxfs/testdir >> /etc/projects
    ```

3. Create an entry in the `/etc/projid` file that maps a project name to the project ID:

    ```
    # echo project_name:project_ID >> /etc/projid
    ```

    For example, to map the project name `testproj` to the project with ID 51:

    ```
    # echo testproj:51 >> /etc/projid
    ```

4. Use the `project` subcommand of `xfs_quota` to define a managed tree in the XFS file system for the project:

    ```
    # xfs_quota -x -c 'project -s project_name' mountpoint
    ```

    For example, to define a managed tree in the `/myxfs` file system for the project `testproj`, which corresponds to the directory hierarchy `/myxfs/testdir`:

    ```
    # xfs_quota -x -c 'project -s testproj' /myxfs
    ```

5. Use the `limit` subcommand to set limits on the disk usage of the project:

    ```
    # xfs_quota -x -c 'limit -p arguments project_name' mountpoint
    ```

    For example, to set a hard limit of 10 GB of disk space for the project `testproj`:

    ```
    # xfs_quota -x -c 'limit -p bhard=10g testproj' /myxfs
    ```

For more information, see the `projects(5)`, `projid(5)`, and `xfs_quota(8)` manual pages.

21.24 Backing up and Restoring XFS File Systems

The `xfsdump` package contains the `xfsdump` and `xfsrestore` utilities. `xfsdump` examines the files in an XFS file system, determines which files need to be backed up, and copies them to the storage medium. Any backups that you create using `xfsdump` are portable between systems with different endian architectures. `xfsrestore` restores a full or incremental backup of an XFS file system. You can also restore individual files and directory hierarchies from backups.

> **Note**
>
> Unlike an LVM snapshot, which immediately creates a sparse clone of a volume, `xfsdump` takes time to make a copy of the file system data.

You can use the `xfsdump` command to create a backup of an XFS file system on a device such as a tape drive, or in a backup file on a different file system. A backup can span multiple physical media that are written on the same device, and you can write multiple backups to the same medium. You can write only a single backup to a file. The command does not overwrite existing XFS backups that it finds on physical media. You must use the appropriate command to erase a physical medium if you need to overwrite any existing backups.

For example, the following command writes a level 0 (base) backup of the XFS file system, `/myxfs` to the device `/dev/st0` and assigns a session label to the backup:

```
# xfsdump -l 0 -L "Backup level 0 of /myxfs `date`" -f /dev/st0 /myxfs
```

You can make incremental dumps relative to an existing backup by using the command:

```
# xfsdump -l level -L "Backup level level of /myxfs `date`" -f /dev/st0 /myxfs
```

A level 1 backup records only file system changes since the level 0 backup, a level 2 backup records only the changes since the latest level 1 backup, and so on up to level 9.

If you interrupt a backup by typing `Ctrl-C` and you did not specify the `-J` option (suppress the dump inventory) to `xfsdump`, you can resume the dump at a later date by specifying the `-R` option:

```
# xfsdump -R -l 1 -L "Backup level 1 of /myxfs `date`" -f /dev/st0 /myxfs
```

In this example, the backup session label from the earlier, interrupted session is overridden.

You use the `xfsrestore` command to find out information about the backups you have made of an XFS file system or to restore data from a backup.

The `xfsrestore -I` command displays information about the available backups, including the session ID and session label. If you want to restore a specific backup session from a backup medium, you can specify either the session ID or the session label.

For example, to restore an XFS file system from a level 0 backup by specifying the session ID:

```
# xfsrestore -f /dev/st0 -S c76b3156-c37c-5b6e-7564-a0963ff8ca8f /myxfs
```

If you specify the `-r` option, you can cumulatively recover all data from a level 0 backup and the higher-level backups that are based on that backup:

```
# xfsrestore -r -f /dev/st0 -v silent /myxfs
```

The command searches the archive looking for backups based on the level 0 backup, and prompts you to choose whether you want to restore each backup in turn. After restoring the backup that you select, the command exits. You must run this command multiple times, first selecting to restore the level 0 backup, and then subsequent higher-level backups up to and including the most recent one that you require to restore the file system data.

> **Note**
>
> After completing a cumulative restoration of an XFS file system, you should delete the `housekeeping` directory that `xfsrestore` creates in the destination directory.

You can recover a selected file or subdirectory contents from the backup medium, as shown in the following example, which recovers the contents of `/myxfs/profile/examples` to `/tmp/profile/examples` from the backup with a specified session label:

```
# xfsrestore -f /dev/sr0 -L "Backup level 0 of /myxfs Sat Mar 2 14:47:59 GMT 2013" \
  -s profile/examples /usr/tmp
```

Alternatively, you can interactively browse a backup by specifying the `-i` option:

```
# xfsrestore -f /dev/sr0 -i
```

This form of the command allows you browse a backup as though it were a file system. You can change directories, list files, add files, delete files, or extract files from a backup.

To copy the entire contents of one XFS file system to another, you can combine `xfsdump` and `xfsrestore`, using the `-J` option to suppress the usual dump inventory housekeeping that the commands perform:

```
# xfsdump -J - /myxfs | xfsrestore -J - /myxfsclone
```

For more information, see the `xfsdump(8)` and `xfsrestore(8)` manual pages.

21.25 Defragmenting an XFS File System

You can use the `xfs_fsr` command to defragment whole XFS file systems or individual files within an XFS file system. As XFS is an extent-based file system, it is usually unnecessary to defragment a whole file system, and doing so is not recommended.

To defragment an individual file, specify the name of the file as the argument to `xfs_fsr`.

```
# xfs_fsr pathname
```

If you run the `xfs_fsr` command without any options, the command defragments all currently mounted, writeable XFS file systems that are listed in `/etc/mtab`. For a period of two hours, the command passes over each file system in turn, attempting to defragment the top ten percent of files that have the greatest number of extents. After two hours, the command records its progress in the file `/var/tmp/.fsrlast_xfs`, and it resumes from that point if you run the command again.

For more information, see the `xfs_fsr(8)` manual page.

21.26 Checking and Repairing an XFS File System

> **Note**
>
> If you have an Oracle Linux Premier Support account and encounter a problem mounting an XFS file system, send a copy of the `/var/log/messages` file to Oracle Support and wait for advice.

If you cannot mount an XFS file system, you can use the `xfs_check` command to check its consistency. Usually, you would only run this command on the device file of an unmounted file system that you believe has a problem. If `xfs_check` displays any output when you do not run it in verbose mode, the file system has an inconsistency.

```
# xfscheck device
```

If you can mount the file system and you do not have a suitable backup, you can use `xfsdump` to attempt to back up the existing file system data, However, the command might fail if the file system's metadata has become too corrupted.

You can use the `xfs_repair` command to attempt to repair an XFS file system specified by its device file. The command replays the journal log to fix any inconsistencies that might have resulted from the file system not being cleanly unmounted. Unless the file system has an inconsistency, it is usually not necessary to use the command, as the journal is replayed every time that you mount an XFS file system.

```
# xfs_repair device
```

If the journal log has become corrupted, you can reset the log by specifying the `-L` option to `xfs_repair`.

Warning

Resetting the log can leave the file system in an inconsistent state, resulting in data loss and data corruption. Unless you are experienced in debugging and repairing XFS file systems using `xfs_db`, it is recommended that you instead recreate the file system and restore its contents from a backup.

If you cannot mount the file system or you do not have a suitable backup, running `xfs_repair` is the only viable option unless you are experienced in using `xfs_db`.

`xfs_db` provides an internal command set that allows you to debug and repair an XFS file system manually. The commands allow you to perform scans on the file system, and to navigate and display its data structures. If you specify the `-x` option to enable expert mode, you can modify the data structures.

```
# xfs_db [-x] device
```

For more information, see the `xfs_check(8)`, `xfs_db(8)` and `xfs_repair(8)` manual pages, and the `help` command within `xfs_db`.

Chapter 22 Shared File System Administration

Table of Contents

This chapter describes administration tasks for the NFS and Samba shared file systems.

22.1 About Shared File Systems

Oracle Linux supports the following shared file system types:

NFS   The Network File System (NFS) is a distributed file system that allows a client computer to access files over a network as though the files were on local storage. See Section 22.2, "About NFS".

Samba   Samba enables the provision of file and print services for Microsoft Windows clients and can integrate with a Windows workgroup, NT4 domain, or Active Directory domain. See Section 22.3, "About Samba".

22.2 About NFS

A Network File System (NFS) server can share directory hierarchies in its local file systems with remote client systems over an IP-based network. After an NFS server exports a directory, NFS clients mount this directory if they have been granted permission to do so. The directory appears to the client systems as if it were a local directory. NFS centralizes storage provisioning and can improves data consistency and reliability.

Oracle Linux 7 supports the following versions of the NFS protocol:

• NFS version 3 (NFSv3), specified in RFC 1813.

• NFS version 4 (NFSv4), specified in RFC 3530.

NFSv3 relies on Remote Procedure Call (RPC) services, which are controlled by the rpcbind service. rpcbind responds to requests for an RPC service and sets up connections for the requested service. In addition, separate services are used to handle locking and mounting protocols. Configuring a firewall to cope with the various ranges of ports that are used by all these services can be complex and error prone.

NFSv4 does not use rpcbind as the NFS server itself listens on TCP port 2049 for service requests. The mounting and locking protocols are also integrated into the NFSv4 protocol, so separate services are also not required for these protocols. These refinements mean that firewall configuration for NFSv4 is no more difficult than for a service such as HTTP.

22.2.1 Configuring an NFS Server

To configure an NFS server:

1. Install the `nfs-utils` package:

   ```
   # yum install nfs-utils
   ```

2. Edit the `/etc/exports` file to define the directories that the server will make available for clients to mount, for example:

   ```
   /var/folder 192.0.2.102(rw,async)
   /usr/local/apps *(all-squash,anonuid=501,anongid=501,ro)
   /var/projects/proj1 192.168.1.0/24(ro) mgmtpc(rw)
   ```

   Each entry consists of the local path to the exported directory, followed by a list of clients that can mount the directory with client-specific mount options in parentheses. If this example:

   - The client system with the IP address 192.0.2.102 can mount `/var/folder` with read and write permissions. All writes to the disk are asynchronous, which means that the server does not wait for write requests to be written to disk before responding to further requests from the client.

   - All clients can mount `/usr/local/apps` read-only, and all connecting users including `root` are mapped to the local unprivileged user with UID 501 and GID 501.

   - All clients on the 192.168.1.0 subnet can mount `/var/projects/proj1` read-only, and the client system named `mgmtpc` can mount the directory with read-write permissions.

   > **Note**
   >
   > There is no space between a client specifier and the parenthesized list of options.

   For more information, see the `exports(5)` manual page.

3. Start the `nfs-server` service, and configure the service to start following a system reboot:

   ```
   # systemctl start nfs-server
   # systemctl enable nfs-server
   ```

4. If the server will serve NFSv4 clients, edit `/etc/idmapd.conf` and edit the definition for the Domain parameter to specify the DNS domain name of the server, for example:

   ```
   Domain = mydom.com
   ```

   This setting prevents the owner and group being unexpectedly listed as the anonymous user or group (`nobody` or `nogroup`) on NFS clients when the `all_squash` mount option has not been specified.

5. If you need to allow access through the firewall for NFSv4 clients only, use the following commands:

   ```
   # firewall-cmd --zone=zone --add-service=nfs
   # firewall-cmd --permanent --zone=zone --add-service=nfs
   ```

   This configuration assumes that `rpc.nfsd` listens for client requests on TCP port 2049.

6. If you need to allow access through the firewall for NFSv3 clients as well as NFSv4 clients:

   a. Edit `/etc/sysconfig/nfs` and create port settings for handling network mount requests and status monitoring:

   ```
   # Port rpc.mountd should listen on.
   MOUNTD_PORT=892

   # Port rpc.statd should listen on.
   STATD_PORT=662
   ```

The port values shown in this example are the default settings that are commented-out in the file.

b. Edit `/etc/sysctl.conf` and configure settings for the TCP and UDP ports on which the network lock manager should listen:

```
fs.nfs.nlm_tcpport = 32803
fs.nfs.nlm_udpport = 32769
```

c. To verify that none of the ports that you have specified in `/etc/sysconfig/nfs` or `/etc/sysctl.conf` is in use, enter the following commands:

```
# lsof -i tcp:32803
# lsof -i udp:32769
# lsof -i :892
# lsof -i :662
```

If any port is in use, use the `lsof -i` command to determine an unused port and amend the setting in `/etc/sysconfig/nfs` or `/etc/sysctl.conf` as appropriate.

d. Shut down and reboot the server.

```
# systemctl reboot
```

NFS fails to start if one of the specified ports is in use, and reports an error in `/var/log/messages`. Edit `/etc/sysconfig/nfs` or `/etc/sysctl.conf` as appropriate to use a different port number for the service that could not start, and attempt to restart the `nfslock` and `nfs-server` services. You can use the `rpcinfo -p` command to confirm on which ports RPC services are listening.

e. Restart the firewall service and configure the firewall to allow NFSv3 connections:

```
# systemctl restart firewalld
# firewall-cmd --zone=zone \
  --add-port=2049/tcp --add-port=2049/udp \
  --add-port=111/tcp --add-port=111/udp \
  --add-port=32803/tcp --add-port=32769/udp \
  --add-port=892/tcp --add-port=892/udp \
  --add-port=662/tcp --add-port=662/udp
# firewall-cmd --permanent --zone=zone \
  --add-port=2049/tcp --add-port=2049/udp \
  --add-port=111/tcp --add-port=111/udp \
  --add-port=32803/tcp --add-port=32769/udp \
  --add-port=892/tcp --add-port=892/udp \
  --add-port=662/tcp --add-port=662/udp
```

The port values shown in this example assume that the default port settings in `/etc/sysconfig/nfs` and `/etc/sysctl.conf` are available for use by RPC services. This configuration also assumes that `rpc.nfsd` and `rpcbind` listen on ports 2049 and 111 respectively.

7. Use the `showmount -e` command to display a list of the exported file systems, for example:

```
# showmount -e
Export list for host01.mydom.com
/var/folder 192.0.2.102
/usr/local/apps *
/var/projects/proj1 192.168.1.0/24 mgmtpc
```

`showmount -a` lists the current clients and the file systems that they have mounted, for example:

```
# showmount -a
mgmtpc.mydom.com:/var/projects/proj1
```

> **Note**
>
> To be able to use the `showmount` command from NFSv4 clients, `MOUNTD_PORT` must be defined in `/etc/sysconfig/nfs` and a firewall rule must allow access on this TCP port.

If you want to export or unexport directories without editing `/etc/exports` and restarting the NFS service, use the `exportfs` command. The following example makes `/var/dev` available with read and write access by all clients, and ignores any existing entries in `/etc/exports`.

```
# exportfs -i -o ro *:/var/dev
```

For more information, see the `exportfs(8)`, `exports(5)`, and `showmount(8)` manual pages.

22.2.2 Mounting an NFS File System

To mount an NFS file system on a client:

1. Install the `nfs-utils` package:

    ```
    # yum install nfs-utils
    ```

2. Use `showmount -e` to discover what file systems an NFS server exports, for example:

    ```
    # showmount -e host01.mydom.com
    Export list for host01.mydom.com
    /var/folder 192.0.2.102
    /usr/local/apps *
    /var/projects/proj1 192.168.1.0/24 mgmtpc
    ```

3. Use the `mount` command to mount an exported NFS file system on an available mount point:

    ```
    # mount -t nfs -o ro,nosuid host01.mydoc.com:/usr/local/apps /apps
    ```

    This example mounts `/usr/local/apps` exported by `host01.mydoc.com` with read-only permissions on `/apps`. The `nosuid` option prevents remote users from gaining higher privileges by running a `setuid` program.

4. To configure the system to mount an NFS file system at boot time, add an entry for the file system to `/etc/fstab`, for example:

    ```
    host01.mydoc.com:/usr/local/apps      /apps      nfs      ro,nosuid  0 0
    ```

For more information, see the `mount(8)`, `nfs(5)`, and `showmount(8)` manual pages.

22.3 About Samba

Samba is an open-source implementation of the Server Message Block (SMB) protocol that allows Oracle Linux to interoperate with Windows systems as both a server and a client. Samba can share Oracle Linux files and printers with Windows systems, and it enables Oracle Linux users to access files on Windows systems. Samba uses the NetBIOS over TCP/IP protocol that allows computer applications that depend on the NetBIOS API to work on TCP/IP networks.

22.3.1 Configuring a Samba Server

To configure a Samba server:

1. Install the `samba` and `samba-winbind` packages:

```
# yum install samba samba-winbind
```

2. Edit `/etc/samba/smb.conf` and configure the sections to support the required services, for example:

```
[global]
security = ADS
realm = MYDOM.REALM
password server = krbsvr.mydom.com
load printers = yes
printing = cups
printcap name = cups

[printers]
comment = All Printers
path = /var/spool/samba
browseable = no
guest ok = yes
writable = no
printable = yes
printer admin = root, @ntadmins, @smbprintadm

[homes]
comment = User home directories
valid users = @smbusers
browsable = no
writable = yes
guest ok = no

[apps]
comment = Shared /usr/local/apps directory
path = /usr/local/apps
browsable = yes
writable = no
guest ok = yes
```

The `[global]` section contains settings for the Samba server. In this example, the server is assumed to be a member of an Active Directory (AD) domain that is running in native mode. Samba relies on tickets issued by the Kerberos server to authenticate clients who want to access local services.

For more information, see Section 22.3.2, "About Samba Configuration for Windows Workgroups and Domains".

The `[printers]` section specifies support for print services. The `path` parameter specifies the location of a spooling directory that receives print jobs from Windows clients before submitting them to the local print spooler. Samba advertises all locally configured printers on the server.

The `[homes]` section provide a personal share for each user in the `smbusers` group. The settings for `browsable` and `writable` prevent other users from browsing home directories, while allowing full access to valid users.

The [apps] section specifies a share named `apps`, which grants Windows users browsing and read-only permission to the `/usr/local/apps` directory.

3. Configure the system firewall to allow incoming TCP connections to ports 139 and 445, and incoming UDP datagrams on ports 137 and 138:

```
# firewall-cmd --zone=zone \
  --add-port=139/tcp --add-port=445/tcp --add-port=137-138/udp
# firewall-cmd --permanent --zone=zone \
  --add-port=139/tcp --add-port=445/tcp --add-port=137-138/udp
```

Add similar rules for other networks from which Samba clients can connect.

The `nmdb` daemon services NetBIOS Name Service requests on UDP port 137 and NetBIOS Datagram Service requests on UDP port 138.

The `smbd` daemon services NetBIOS Session Service requests on TCP port 139 and Microsoft Directory Service requests on TCP port 445.

4. Start the `smb` service, and configure the service to start following a system reboot:

```
# systemctl start smb
# systemctl enable smb
```

If you change the `/etc/samba/smb.conf` file and any files that it references, the `smb` service will reload its configuration automatically after a delay of up to one minute. You can force `smb` to reload its configuration by sending a `SIGHUP` signal to the service daemon:

```
# killall -SIGHUP smbd
```

Making `smb` reload its configuration has no effect on established connections. You must restart the `smb` service or the existing users of the service must disconnect and then reconnect.

To restart the `smb` service, use the following command:

```
# systemctl restart smb
```

For more information, see the `smb.conf(5)` and `smbd(8)` manual pages and http://www.samba.org/samba/docs/.

22.3.2 About Samba Configuration for Windows Workgroups and Domains

Windows systems on an enterprise network usually belong either to a workgroup or to a domain.

Workgroups are usually only configured on networks that connect a small number of computers. A workgroup environment is a peer-to-peer network where systems do not rely on each other for services and there is no centralized management. User accounts, access control, and system resources are configured independently on each system. Such systems can share resources only if configured to do so.

A Samba server can act as a standalone server within a workgroup.

More typically, corporate networks configure domains to allow large numbers of networked systems to be administered centrally. A domain is a group of trusted computers that share security and access control. Systems known as domain controllers provides centralized management and security. Windows domains are usually configured to use Active Directory (AD), which uses the Lightweight Directory Access Protocol (LDAP) to implement versions of Kerberos and DNS providing authentication, access control to domain resources, and name service. Some Windows domains use Windows NT4 security, which does not use Kerberos to perform authentication.

A Samba server can be a member of an AD or NT4 security domain, but it cannot operate as a domain controller. As domain member Samba server must authenticate itself with a domain controller and so is controlled by the security rules of the domain. The domain controller authenticates clients, and the Samba server controls access to printers and network shares.

22.3.2.1 Configuring Samba as a Standalone Server

A standalone Samba server can be a member of a workgroup. The following `[global]` section from `/etc/samba/smb.conf` shows an example of how to configure a standalone server using share-level security:

```
[global]
security = share
workgroup = workgroup_name
netbios name = netbios_name
```

The client provides only a password and not a user name to the server. Typically, each share is associated with a `valid users` parameter and the server validates the password against the hashed passwords stored in `/etc/passwd`, `/etc/shadow`, NIS, or LDAP for the listed users. Using share-level security is discouraged in favor of user-level security, for example:

```
[global]
security = user
workgroup = workgroup_name
netbios name = netbios_name
```

In the user security model, a client must supply a valid user name and password. This model supports encrypted passwords. If the server successfully validates the client's user name and password, the client can mount multiple shares without being required to specify a password. Use the `smbpasswd` command to create an entry for a user in the Samba password file, for example:

```
# smbpasswd -a guest
New SMB password: password
Retype new SMB password: password
Added user guest.
```

The user must already exist as a user on the system. If a user is permitted to log into the server, he or she can use the `smbpasswd` command to change his or her password.

If a Windows user has a different user name from his or her user name on the Samba server, create a mapping between the names in the `/etc/samba/smbusers` file, for example:

```
root = admin administrator root
nobody = guest nobody pcguest smbguest
eddie = ejones
fiona = fchau
```

The first entry on each line is the user name on the Samba server. The entries after the equals sign (=) are the equivalent Windows user names.

> **Note**
>
> Only the user security model uses Samba passwords.

The server security model, where the Samba server relies on another server to authenticate user names and passwords, is deprecated as it has numerous security and interoperability issues.

22.3.2.2 Configuring Samba as a Member of an ADS Domain

In the Activity Directory Server (ADS) security model, Samba acts as a domain member server in an ADS realm, and clients use Kerberos tickets for Active Directory authentication. You must configure Kerberos and join the server to the domain, which creates a machine account for your server on the domain controller.

To add a Samba server to an Active Directory domain:

1. Edit `/etc/samba/smb.conf` and configure the `[global]` section to use ADS:

   ```
   [global]
   ```

```
security = ADS
realm = KERBEROS.REALM
```

It might also be necessary to specify the password server explicitly if different servers support AD services and Kerberos authentication:

```
password server = kerberos_server.your_domain
```

2. Install the `krb5-server` package:

```
# yum install krb5-server
```

3. Create a Kerberos ticket for the `Administrator` account in the Kerberos domain, for example:

```
# kinit Administrator@MYDOMAIN.COM
```

This command creates the Kerberos ticket that is required to join the server to the AD domain.

4. Join the server to the AD domain:

```
# net ads join -S winads.mydom.com -U Administrator%password
```

In this example, the AD server is `winads.mydom.com` and `password` is the password for the Administrator account.

The command creates a machine account in Active Directory for the Samba server and allows it to join the domain.

5. Restart the `smb` service:

```
# systemctl restart smb
```

22.3.2.3 Configuring Samba as a Member of a Windows NT4 Security Domain

> **Note**
>
> If the Samba server acts as a Primary or Backup Domain Controller, do not use the domain security model. Configure the system as a standalone server that uses the user security model instead. See Section 22.3.2.1, "Configuring Samba as a Standalone Server".

The domain security model is used with domains that implement Windows NT4 security. The Samba server must have a machine account in the domain (a domain security trust account). Samba authenticates user names and passwords with either a primary or a secondary domain controller.

To add a Samba server to an NT4 domain:

1. On the primary domain controller, use the Server Manager to add a machine account for the Samba server.

2. Edit `/etc/samba/smb.conf` and configure the `[global]` section to use ADS:

```
[global]
security = domain
workgroup = DOMAIN
netbios name = SERVERNAME
```

3. Join the server to the domain:

```
# net rpc join -S winpdc.mydom.com -U Administrator%password
```

In this example, the primary domain controller is `winpdc.mydom.com` and *password* is the password for the Administrator account.

4. Restart the `smb` service:

```
# systemctl restart smb
```

5. Create an account for each user who is allowed access to shares or printers:

```
# useradd -s /sbin/nologin username
# passwd username
```

In this example, the account's login shell is set to `/sbin/nologin` to prevent direct logins.

22.3.3 Accessing Samba Shares from a Windows Client

To access a share on a Samba server from Windows, open Computer or Windows Explorer, and enter the host name of the Samba server and the share name using the following format:

```
\\server_name\share_name
```

If you enter `\\server_name`, Windows displays the directories and printers that the server is sharing. You can also use the same syntax to map a network drive to a share name.

22.3.4 Accessing Samba Shares from an Oracle Linux Client

> **Note**
>
> To be able to use the commands described in this section, use `yum` to install the `samba-client` and `cifs-utils` packages.

You can use the `findsmb` command to query a subnet for Samba servers. The command displays the IP address, NetBIOS name, workgroup, operating system and version for each server that it finds.

Alternatively, you can use the `smbtree` command, which is a text-based SMB network browser that displays the hierarchy of known domains, servers in those domains, and shares on those servers.

The GNOME and KDE desktops provide browser-based file managers that you can use to view Windows shares on the network. Enter `smb:` in the location bar of a file manager to browse network shares.

To connect to a Windows share from the command line, use the `smbclient` command:

```
$ smbclient //server_name/share_name [-U username]
```

After logging in, enter `help` at the `smb:\>` prompt to display a list of available commands.

To mount a Samba share, use a command such as the following:

```
# mount -t cifs //server_name/share_name mountpoint -o credentials=credfile
```

where the credentials file contains settings for `username`, `password`, and `domain`, for example:

```
username=eddie
password=clydenw
domain=MYDOMWKG
```

The argument to `domain` can be the name of a domain or a workgroup.

Caution

As the credentials file contains a plain-text password, use `chmod` to make it readable only by you, for example:

```
# chmod 400 credfile
```

If the Samba server is a domain member server in an AD domain and your current login session was authenticated by the Kerberos server in the domain, you can use your existing session credentials by specifying the **sec=krb5** option instead of a credentials file:

```
# mount -t cifs //server_name/share_name mountpoint -o sec=krb5
```

For more information, see the `findsmb(1)`, `mount.cifs(8)`, `smbclient(1)`, and `smbtree(1)` manual pages.

Chapter 23 Oracle Cluster File System Version 2

Table of Contents

This chapter describes how to configure and use the Oracle Cluster File System Version 2 (OCFS2) file system.

23.1 About OCFS2

Oracle Cluster File System version 2 (OCFS2) is a general-purpose, high-performance, high-availability, shared-disk file system intended for use in clusters. It is also possible to mount an OCFS2 volume on a standalone, non-clustered system.

Although it might seem that there is no benefit in mounting `ocfs2` locally as compared to alternative file systems such as `ext4` or `btrfs`, you can use the `reflink` command with OCFS2 to create copy-on-write clones of individual files in a similar way to using the `cp --reflink` command with the btrfs file system. Typically, such clones allow you to save disk space when storing multiple copies of very similar files, such as VM images or Linux Containers. In addition, mounting a local OCFS2 file system allows you to subsequently migrate it to a cluster file system without requiring any conversion. Note that when using the `reflink` command, the resulting filesystem behaves like a clone of the original filesystem. This means that their UUIDs are identical. When using `reflink` to create a clone, you must change the UUID using the `tunefs.ocfs2` command. See Section 23.2.10, "Querying and Changing Volume Parameters" for more information.

Almost all applications can use OCFS2 as it provides local file-system semantics. Applications that are cluster-aware can use cache-coherent parallel I/O from multiple cluster nodes to balance activity across the cluster, or they can use of the available file-system functionality to fail over and run on another node in the event that a node fails. The following examples typify some use cases for OCFS2:

- Oracle VM to host shared access to virtual machine images.

- Oracle VM and VirtualBox to allow Linux guest machines to share a file system.

- Oracle Real Application Cluster (RAC) in database clusters.

- Oracle E-Business Suite in middleware clusters.

OCFS2 has a large number of features that make it suitable for deployment in an enterprise-level computing environment:

- Support for ordered and write-back data journaling that provides file system consistency in the event of power failure or system crash.

- Block sizes ranging from 512 bytes to 4 KB, and file-system cluster sizes ranging from 4 KB to 1 MB (both in increments of powers of 2). The maximum supported volume size is 16 TB, which corresponds to a cluster size of 4 KB. A volume size as large as 4 PB is theoretically possible for a cluster size of 1 MB, although this limit has not been tested.

- Extent-based allocations for efficient storage of very large files.

- Optimized allocation support for sparse files, inline-data, unwritten extents, hole punching, reflinks, and allocation reservation for high performance and efficient storage.

- Indexing of directories to allow efficient access to a directory even if it contains millions of objects.

- Metadata checksums for the detection of corrupted inodes and directories.

- Extended attributes to allow an unlimited number of `name:value` pairs to be attached to file system objects such as regular files, directories, and symbolic links.

- Advanced security support for POSIX ACLs and SELinux in addition to the traditional file-access permission model.

- Support for user and group quotas.

- Support for heterogeneous clusters of nodes with a mixture of 32-bit and 64-bit, little-endian (x86, x86_64, ia64) and big-endian (ppc64) architectures.

- An easy-to-configure, in-kernel cluster-stack (O2CB) with a distributed lock manager (DLM), which manages concurrent access from the cluster nodes.

- Support for buffered, direct, asynchronous, splice and memory-mapped I/O.

- A tool set that uses similar parameters to the `ext3` file system.

23.2 Installing and Configuring OCFS2

The procedures in the following sections describe how to set up a cluster to use OCFS2.

- Section 23.2.1, "Preparing a Cluster for OCFS2"

- Section 23.2.2, "Configuring the Firewall"

- Section 23.2.3, "Configuring the Cluster Software"

- Section 23.2.4, "Creating the Configuration File for the Cluster Stack"

- Section 23.2.5, "Configuring the Cluster Stack"

- Section 23.2.6, "Configuring the Kernel for Cluster Operation"

23.2.1 Preparing a Cluster for OCFS2

For best performance, each node in the cluster should have at least two network interfaces. One interface is connected to a public network to allow general access to the systems. The other interface is used for private communication between the nodes; the *cluster heartbeat* that determines how the cluster nodes coordinate their access to shared resources and how they monitor each other's state. These interface must be connected via a network switch. Ensure that all network interfaces are configured and working before continuing to configure the cluster.

You have a choice of two cluster heartbeat configurations:

• Local heartbeat thread for each shared device. In this mode, a node starts a heartbeat thread when it mounts an OCFS2 volume and stops the thread when it unmounts the volume. This is the default heartbeat mode. There is a large CPU overhead on nodes that mount a large number of OCFS2 volumes as each mount requires a separate heartbeat thread. A large number of mounts also increases the risk of a node fencing itself out of the cluster due to a heartbeat I/O timeout on a single mount.

• Global heartbeat on specific shared devices. You can configure any OCFS2 volume as a global heartbeat device provided that it occupies a whole disk device and not a partition. In this mode, the heartbeat to the device starts when the cluster comes online and stops when the cluster goes offline. This mode is recommended for clusters that mount a large number of OCFS2 volumes. A node fences itself out of the cluster if a heartbeat I/O timeout occurs on more than half of the global heartbeat devices. To provide redundancy against failure of one of the devices, you should therefore configure at least three global heartbeat devices.

Figure 23.1 shows a cluster of four nodes connected via a network switch to a LAN and a network storage server. The nodes and the storage server are also connected via a switch to a private network that they use for the local cluster heartbeat.

Figure 23.1 Cluster Configuration Using a Private Network
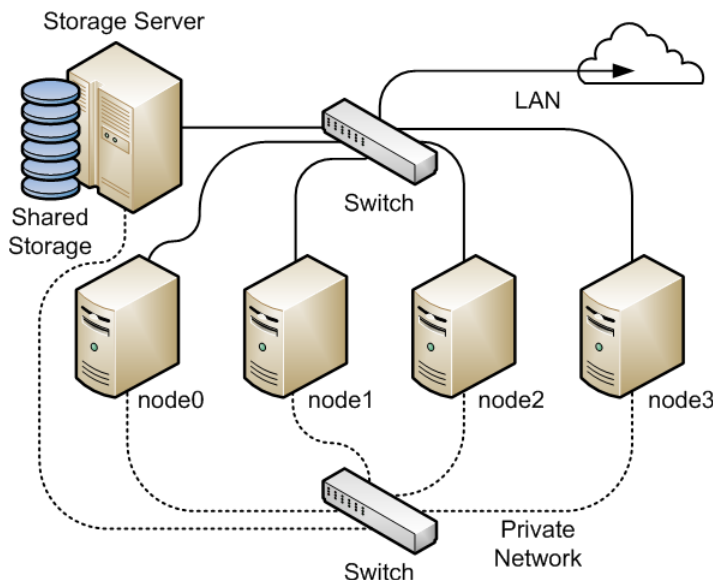
It is possible to configure and use OCFS2 without using a private network but such a configuration increases the probability of a node fencing itself out of the cluster due to an I/O heartbeat timeout.

23.2.2 Configuring the Firewall

Configure or disable the firewall on each node to allow access on the interface that the cluster will use for private cluster communication. By default, the cluster uses both TCP and UDP over port 7777.

To allow incoming TCP connections and UDP datagrams on port 7777, use the following commands:

```
# firewall-cmd --zone=zone --add-port=7777/tcp --add-port=7777/udp
# firewall-cmd --permanent --zone=zone --add-port=7777/tcp --add-port=7777/udp
```

23.2.3 Configuring the Cluster Software

Ideally, each node should be running the same version of the OCFS2 software and a compatible version of the Oracle Linux Unbreakable Enterprise Kernel (UEK). It is possible for a cluster to run with mixed versions of the OCFS2 and UEK software, for example, while you are performing a rolling update of a cluster. The cluster node that is running the lowest version of the software determines the set of usable features.

Use `yum` to install or upgrade the following packages to the same version on each node:

- `kernel-uek`

- `ocfs2-tools`

> **Note**
>
> If you want to use the global heartbeat feature, you must install `ocfs2-tools-1.8.0-11` or later.

23.2.4 Creating the Configuration File for the Cluster Stack

You can create the configuration file by using the `o2cb` command or a text editor.

To configure the cluster stack by using the `o2cb` command:

1. Use the following command to create a cluster definition.

   ```
   # o2cb add-cluster cluster_name
   ```

   For example, to define a cluster named `mycluster` with four nodes:

   ```
   # o2cb add-cluster mycluster
   ```

   The command creates the configuration file `/etc/ocfs2/cluster.conf` if it does not already exist.

2. For each node, use the following command to define the node.

   ```
   # o2cb add-node cluster_name node_name --ip ip_address
   ```

   The name of the node must be same as the value of system's `HOSTNAME` that is configured in `/etc/sysconfig/network`. The IP address is the one that the node will use for private communication in the cluster.

   For example, to define a node named `node0` with the IP address 10.1.0.100 in the cluster `mycluster`:

   ```
   # o2cb add-node mycluster node0 --ip 10.1.0.100
   ```

3. If you want the cluster to use global heartbeat devices, use the following commands.

   ```
   # o2cb add-heartbeat cluster_name device1
   .
   ```

```
.
.
# o2cb heartbeat-mode cluster_name global
```

> **Note**
>
> You must configure global heartbeat to use whole disk devices. You cannot configure a global heartbeat device on a disk partition.

For example, to use `/dev/sdd`, `/dev/sdg`, and `/dev/sdj` as global heartbeat devices:

```
# o2cb add-heartbeat mycluster /dev/sdd
# o2cb add-heartbeat mycluster /dev/sdg
# o2cb add-heartbeat mycluster /dev/sdj
# o2cb heartbeat-mode mycluster global
```

4. Copy the cluster configuration file `/etc/ocfs2/cluster.conf` to each node in the cluster.

> **Note**
>
> Any changes that you make to the cluster configuration file do not take effect until you restart the cluster stack.

The following sample configuration file `/etc/ocfs2/cluster.conf` defines a 4-node cluster named `mycluster` with a local heartbeat.

```
node:
 name = node0
 cluster = mycluster
 number = 0
 ip_address = 10.1.0.100
 ip_port = 7777

node:
        name = node1
        cluster = mycluster
        number = 1
        ip_address = 10.1.0.101
        ip_port = 7777

node:
        name = node2
        cluster = mycluster
        number = 2
        ip_address = 10.1.0.102
        ip_port = 7777

node:
        name = node3
        cluster = mycluster
        number = 3
        ip_address = 10.1.0.103
        ip_port = 7777

cluster:
        name = mycluster
        heartbeat_mode = local
        node_count = 4
```

If you configure your cluster to use a global heartbeat, the file also include entries for the global heartbeat devices.

```
node:
        name = node0
```

```
        cluster = mycluster
        number = 0
        ip_address = 10.1.0.100
        ip_port = 7777

node:
        name = node1
        cluster = mycluster
        number = 1
        ip_address = 10.1.0.101
        ip_port = 7777

node:
        name = node2
        cluster = mycluster
        number = 2
        ip_address = 10.1.0.102
        ip_port = 7777

node:
        name = node3
        cluster = mycluster
        number = 3
        ip_address = 10.1.0.103
        ip_port = 7777

cluster:
        name = mycluster
        heartbeat_mode = global
        node_count = 4

heartbeat:
        cluster = mycluster
        region = 7DA5015346C245E6A41AA85E2E7EA3CF

heartbeat:
        cluster = mycluster
        region = 4F9FBB0D9B6341729F21A8891B9A05BD

heartbeat:
        cluster = mycluster
        region = B423C7EEE9FC426790FC411972C91CC3
```

The cluster heartbeat mode is now shown as `global`, and the heartbeat regions are represented by the UUIDs of their block devices.

If you edit the configuration file manually, ensure that you use the following layout:

- The `cluster:`, `heartbeat:`, and `node:` headings must start in the first column.

- Each parameter entry must be indented by one tab space.

- A blank line must separate each section that defines the cluster, a heartbeat device, or a node.

23.2.5 Configuring the Cluster Stack

To configure the cluster stack:

1. Run the following command on each node of the cluster:

   ```
   # /sbin/o2cb.init configure
   ```

   The following table describes the values for which you are prompted.

Prompt	Description
`Load O2CB driver on boot (y/n)`	Whether the cluster stack driver should be loaded at boot time. The default response is `n`.
`Cluster stack backing O2CB`	The name of the cluster stack service. The default and usual response is `o2cb`.
`Cluster to start at boot (Enter "none" to clear)`	Enter the name of your cluster that you defined in the cluster configuration file, `/etc/ocfs2/cluster.conf`.
`Specify heartbeat dead threshold (>=7)`	The number of 2-second heartbeats that must elapse without response before a node is considered dead. To calculate the value to enter, divide the required threshold time period by 2 and add 1. For example, to set the threshold time period to 120 seconds, enter a value of 61. The default value is 31, which corresponds to a threshold time period of 60 seconds. **Note** If your system uses multipathed storage, the recommended value is 61 or greater.
`Specify network idle timeout in ms (>=5000)`	The time in milliseconds that must elapse before a network connection is considered dead. The default value is 30,000 milliseconds. **Note** For bonded network interfaces, the recommended value is 30,000 milliseconds or greater.
`Specify network keepalive delay in ms (>=1000)`	The maximum delay in milliseconds between sending keepalive packets to another node. The default and recommended value is 2,000 milliseconds.
`Specify network reconnect delay in ms (>=2000)`	The minimum delay in milliseconds between reconnection attempts if a network connection goes down. The default and recommended value is 2,000 milliseconds.

To verify the settings for the cluster stack, enter the `/sbin/o2cb.init status` command:

```
# /sbin/o2cb.init status
Driver for "configfs": Loaded
Filesystem "configfs": Mounted
Stack glue driver: Loaded
Stack plugin "o2cb": Loaded
Driver for "ocfs2_dlmfs": Loaded
Filesystem "ocfs2_dlmfs": Mounted
Checking O2CB cluster "mycluster": Online
  Heartbeat dead threshold: 61
  Network idle timeout: 30000
  Network keepalive delay: 2000
  Network reconnect delay: 2000
  Heartbeat mode: Local
Checking O2CB heartbeat: Active
```

In this example, the cluster is online and is using local heartbeat mode. If no volumes have been configured, the O2CB heartbeat is shown as `Not active` rather than `Active`.

The next example shows the command output for an online cluster that is using three global heartbeat devices:

```
# /sbin/o2cb.init status
Driver for "configfs": Loaded
Filesystem "configfs": Mounted
Stack glue driver: Loaded
Stack plugin "o2cb": Loaded
Driver for "ocfs2_dlmfs": Loaded
Filesystem "ocfs2_dlmfs": Mounted
Checking O2CB cluster "mycluster": Online
  Heartbeat dead threshold: 61
  Network idle timeout: 30000
  Network keepalive delay: 2000
  Network reconnect delay: 2000
  Heartbeat mode: Global
Checking O2CB heartbeat: Active
  7DA5015346C245E6A41AA85E2E7EA3CF  /dev/sdd
  4F9FBB0D9B6341729F21A8891B9A05BD  /dev/sdg
  B423C7EEE9FC426790FC411972C91CC3  /dev/sdj
```

2. Configure the `o2cb` and `ocfs2` services so that they start at boot time after networking is enabled:

```
# systemctl enable o2cb
# systemctl enable ocfs2
```

These settings allow the node to mount OCFS2 volumes automatically when the system starts.

23.2.6 Configuring the Kernel for Cluster Operation

For the correct operation of the cluster, you must configure the kernel settings shown in the following table:

Kernel Setting	Description
panic	Specifies the number of seconds after a panic before a system will automatically reset itself.
	If the value is 0, the system hangs, which allows you to collect detailed information about the panic for troubleshooting. This is the default value.
	To enable automatic reset, set a non-zero value. If you require a memory image (`vmcore`), allow enough time for Kdump to create this image. The suggested value is 30 seconds, although large systems will require a longer time.
panic_on_oops	Specifies that a system must panic if a kernel oops occurs. If a kernel thread required for cluster operation crashes, the system must reset itself. Otherwise, another node might not be able to tell whether a node is slow to respond or unable to respond, causing cluster operations to hang.

On each node, enter the following commands to set the recommended values for `panic` and `panic_on_oops`:

```
# sysctl kernel.panic = 30
# sysctl kernel.panic_on_oops = 1
```

To make the change persist across reboots, add the following entries to the `/etc/sysctl.conf` file:

```
# Define panic and panic_on_oops for cluster operation
kernel.panic = 30
kernel.panic_on_oops = 1
```

23.2.7 Starting and Stopping the Cluster Stack

The following table shows the commands that you can use to perform various operations on the cluster stack.

Command	Description
/sbin/o2cb.init status	Check the status of the cluster stack.
/sbin/o2cb.init online	Start the cluster stack.
/sbin/o2cb.init offline	Stop the cluster stack.
/sbin/o2cb.init unload	Unload the cluster stack.

23.2.8 Creating OCFS2 volumes

You can use the mkfs.ocfs2 command to create an OCFS2 volume on a device. If you want to label the volume and mount it by specifying the label, the device must correspond to a partition. You cannot mount an unpartitioned disk device by specifying a label. The following table shows the most useful options that you can use when creating an OCFS2 volume.

Command Option	Description
-b block-size --block-size block-size	Specifies the unit size for I/O transactions to and from the file system, and the size of inode and extent blocks. The supported block sizes are 512 (512 bytes), 1K, 2K, and 4K. The default and recommended block size is 4K (4 kilobytes).
-C cluster-size --cluster-size cluster-size	Specifies the unit size for space used to allocate file data. The supported cluster sizes are 4K, 8K, 16K, 32K, 64K, 128K, 256K, 512K, and 1M (1 megabyte). The default cluster size is 4K (4 kilobytes).
--fs-feature-level=feature-level	Allows you select a set of file-system features: default — Enables support for the sparse files, unwritten extents, and inline data features. max-compat — Enables only those features that are understood by older versions of OCFS2. max-features — Enables all features that OCFS2 currently supports.
--fs_features=feature	Allows you to enable or disable individual features such as support for sparse files, unwritten extents, and backup superblocks. For more information, see the mkfs.ocfs2(8) manual page.
-J size=journal-size --journal-options size=journal-size	Specifies the size of the write-ahead journal. If not specified, the size is determined from the file system usage type that you specify to the -T option, and, otherwise, from the volume size. The default size of the

Command Option	Description
	journal is 64M (64 MB) for `datafiles`, 256M (256 MB) for `mail`, and 128M (128 MB) for `vmstore`.

`-L volume-label`
 	
`--label volume-label`
	Specifies a descriptive name for the volume that allows you to identify it easily on different cluster nodes.

`-N number`
 	
`--node-slots number`
	Determines the maximum number of nodes that can concurrently access a volume, which is limited by the number of node slots for system files such as the file-system journal. For best performance, set the number of node slots to at least twice the number of nodes. If you subsequently increase the number of node slots, performance can suffer because the journal will no longer be contiguously laid out on the outer edge of the disk platter.

`-T file-system-usage-type`	Specifies the type of usage for the file system:
	`datafiles` — Database files are typically few in number, fully allocated, and relatively large. Such files require few metadata changes, and do not benefit from having a large journal.
	`mail` — Mail server files are typically many in number, and relatively small. Such files require many metadata changes, and benefit from having a large journal.
	`vmstore` — Virtual machine image files are typically few in number, sparsely allocated, and relatively large. Such files require a moderate number of metadata changes and a medium sized journal.

For example, create an OCFS2 volume on `/dev/sdc1` labeled as `myvol` using all the default settings for generic usage on file systems that are no larger than a few gigabytes. The default values are a 4 KB block and cluster size, eight node slots, a 256 MB journal, and support for default file-system features.

```
# mkfs.ocfs2 -L "myvol" /dev/sdc1
```

Create an OCFS2 volume on `/dev/sdd2` labeled as `dbvol` for use with database files. In this case, the cluster size is set to 128 KB and the journal size to 32 MB.

```
# mkfs.ocfs2 -L "dbvol" -T datafiles /dev/sdd2
```

Create an OCFS2 volume on `/dev/sde1` with a 16 KB cluster size, a 128 MB journal, 16 node slots, and support enabled for all features except refcount trees.

```
# mkfs.ocfs2 -C 16K -J size=128M -N 16 --fs-feature-level=max-features \
  --fs-features=norefcount /dev/sde1
```

Note

Do not create an OCFS2 volume on an LVM logical volume. LVM is not cluster-aware.

You cannot change the block and cluster size of an OCFS2 volume after it has been created. You can use the `tunefs.ocfs2` command to modify other settings for the file system with certain restrictions. For more information, see the `tunefs.ocfs2(8)` manual page.

If you intend the volume to store database files, do not specify a cluster size that is smaller than the block size of the database.

The default cluster size of 4 KB is not suitable if the file system is larger than a few gigabytes. The following table suggests minimum cluster size settings for different file system size ranges:

File System Size	Suggested Minimum Cluster Size
1 GB - 10 GB	8K
10GB - 100 GB	16K
100 GB - 1 TB	32K
1 TB - 10 TB	64K
10 TB - 16 TB	128K

23.2.9 Mounting OCFS2 Volumes

As shown in the following example, specify the `_netdev` option in `/etc/fstab` if you want the system to mount an OCFS2 volume at boot time after networking is started, and to unmount the file system before networking is stopped.

```
myocfs2vol  /dbvol1  ocfs2     _netdev,defaults  0 0
```

Note

The file system will not mount unless you have enabled the `o2cb` and `ocfs2` services to start after networking is started. See Section 23.2.5, "Configuring the Cluster Stack".

23.2.10 Querying and Changing Volume Parameters

You can use the `tunefs.ocfs2` command to query or change volume parameters. For example, to find out the label, UUID and the number of node slots for a volume:

```
# tunefs.ocfs2 -Q "Label = %V\nUUID = %U\nNumSlots =%N\n" /dev/sdb
Label = myvol
UUID = CBB8D5E0C169497C8B52A0FD555C7A3E
NumSlots = 4
```

Generate a new UUID for a volume:

```
# tunefs.ocfs2 -U /dev/sda
# tunefs.ocfs2 -Q "Label = %V\nUUID = %U\nNumSlots =%N\n" /dev/sdb
Label = myvol
UUID = 48E56A2BBAB34A9EB1BE832B3C36AB5C
NumSlots = 4
```

23.3 Troubleshooting OCFS2

The following sections describes some techniques that you can use for investigating any problems that you encounter with OCFS2.

23.3.1 Recommended Tools for Debugging

To you want to capture an oops trace, it is recommended that you set up `netconsole` on the nodes.

If you want to capture the DLM's network traffic between the nodes, you can use tcpdump. For example, to capture TCP traffic on port 7777 for the private network interface `em2`, you could use a command such as the following:

```
# tcpdump -i em2 -C 10 -W 15 -s 10000 -Sw /tmp/`hostname -s`_tcpdump.log \
  -ttt 'port 7777' &
```

You can use the `debugfs.ocfs2` command, which is similar in behavior to the `debugfs` command for the `ext3` file system, and allows you to trace events in the OCFS2 driver, determine lock statuses, walk directory structures, examine inodes, and so on.

For more information, see the `debugfs.ocfs2(8)` manual page.

The `o2image` command saves an OCFS2 file system's metadata (including information about inodes, file names, and directory names) to an image file on another file system. As the image file contains only metadata, it is much smaller than the original file system. You can use `debugfs.ocfs2` to open the image file, and analyze the file system layout to determine the cause of a file system corruption or performance problem.

For example, the following command creates the image `/tmp/sda2.img` from the OCFS2 file system on the device `/dev/sda2`:

```
# o2image /dev/sda2 /tmp/sda2.img
```

For more information, see the `o2image(8)` manual page.

23.3.2 Mounting the debugfs File System

OCFS2 uses the `debugfs` file system to allow access from user space to information about its in-kernel state. You must mount the `debugfs` file system to be able to use the `debugfs.ocfs2` command.

To mount the `debugfs` file system, add the following line to `/etc/fstab`:

```
debugfs    /sys/kernel/debug    debugfs  defaults  0 0
```

and run the `mount -a` command.

23.3.3 Configuring OCFS2 Tracing

The following table shows some of the commands that are useful for tracing problems in OCFS2.

Command	Description
`debugfs.ocfs2 -l`	List all trace bits and their statuses.
`debugfs.ocfs2 -l SUPER allow`	Enable tracing for the superblock.
`debugfs.ocfs2 -l SUPER off`	Disable tracing for the superblock.
`debugfs.ocfs2 -l SUPER deny`	Disallow tracing for the superblock, even if implicitly enabled by another tracing mode setting.

`debugfs.ocfs2 -l HEARTBEAT \`
 	
`ENTRY EXIT allow`
	Enable heartbeat tracing.

`debugfs.ocfs2 -l HEARTBEAT off \`
 	
`ENTRY EXIT deny`
	Disable heartbeat tracing. `ENTRY` and `EXIT` are set to `deny` as they exist in all trace paths.

`debugfs.ocfs2 -l ENTRY EXIT \`
 	
`NAMEI INODE allow`
	Enable tracing for the file system.

`debugfs.ocfs2 -l ENTRY EXIT \`
 	
`deny NAMEI INODE allow`
	Disable tracing for the file system.

`debugfs.ocfs2 -l ENTRY EXIT \`
 	
`DLM DLM_THREAD allow`
	Enable tracing for the DLM.

`debugfs.ocfs2 -l ENTRY EXIT \`
 	
`deny DLM DLM_THREAD allow`
	Disable tracing for the DLM.

One method for obtaining a trace its to enable the trace, sleep for a short while, and then disable the trace. As shown in the following example, to avoid seeing unnecessary output, you should reset the trace bits to their default settings after you have finished.

```
# debugfs.ocfs2 -l ENTRY EXIT NAMEI INODE allow && sleep 10 && \
  debugfs.ocfs2 -l ENTRY EXIT deny NAMEI INODE off
```

To limit the amount of information displayed, enable only the trace bits that you believe are relevant to understanding the problem.

If you believe a specific file system command, such as `mv`, is causing an error, the following example shows the commands that you can use to help you trace the error.

```
# debugfs.ocfs2 -l ENTRY EXIT NAMEI INODE allow
# mv source destination & CMD_PID=$(jobs -p %-)
# echo $CMD_PID
# debugfs.ocfs2 -l ENTRY EXIT deny NAMEI INODE off
```

As the trace is enabled for all mounted OCFS2 volumes, knowing the correct process ID can help you to interpret the trace.

For more information, see the `debugfs.ocfs2(8)` manual page.

23.3.4 Debugging File System Locks

If an OCFS2 volume hangs, you can use the following steps to help you determine which locks are busy and the processes that are likely to be holding the locks.

1. Mount the debug file system.

```
# mount -t debugfs debugfs /sys/kernel/debug
```

2. Dump the lock statuses for the file system device (`/dev/sdx1` in this example).

```
# echo "fs_locks" | debugfs.ocfs2 /dev/sdx1 >/tmp/fslocks 62
Lockres: M00000000000006672078b84822 Mode: Protected Read
Flags: Initialized Attached
RO Holders: 0 EX Holders: 0
Pending Action: None Pending Unlock Action: None
Requested Mode: Protected Read Blocking Mode: Invalid
```

The `Lockres` field is the lock name used by the DLM. The lock name is a combination of a lock-type identifier, an inode number, and a generation number. The following table shows the possible lock types.

Identifier	Lock Type
D	File data.
M	Metadata.
R	Rename.
S	Superblock.
W	Read-write.

3. Use the `Lockres` value to obtain the inode number and generation number for the lock.

```
# echo "stat <M00000000000006672078b84822>" | debugfs.ocfs2 -n /dev/sdx1
Inode: 419616    Mode: 0666    Generation: 2025343010 (0x78b84822)
...
```

4. Determine the file system object to which the inode number relates by using the following command.

```
# echo "locate <419616>" | debugfs.ocfs2 -n /dev/sdx1
419616 /linux-2.6.15/arch/i386/kernel/semaphore.c
```

5. Obtain the lock names that are associated with the file system object.

```
# echo "encode /linux-2.6.15/arch/i386/kernel/semaphore.c" | \
  debugfs.ocfs2 -n /dev/sdx1
M00000000000006672078b84822 D00000000000006672078b84822 W00000000000006672078b84822
```

In this example, a metadata lock, a file data lock, and a read-write lock are associated with the file system object.

6. Determine the DLM domain of the file system.

```
# echo "stats" | debugfs.ocfs2 -n /dev/sdX1 | grep UUID: | while read a b ; do echo $b ; done
82DA8137A49A47E4B187F74E09FBBB4B
```

7. Use the values of the DLM domain and the lock name with the following command, which enables debugging for the DLM.

```
# echo R 82DA8137A49A47E4B187F74E09FBBB4B \
  M00000000000006672078b84822 > /proc/fs/ocfs2_dlm/debug
```

8. Examine the debug messages.

```
# dmesg | tail
struct dlm_ctxt: 82DA8137A49A47E4B187F74E09FBBB4B, node=3, key=965960985
  lockres: M00000000000006672078b84822, owner=1, state=0 last used: 0,
  on purge list: no granted queue:
```

```
     type=3, conv=-1, node=3, cookie=11673330234144325711, ast=(empty=y,pend=n),
     bast=(empty=y,pend=n)
converting queue:
blocked queue:
```

The DLM supports 3 lock modes: no lock (`type=0`), protected read (`type=3`), and exclusive (`type=5`). In this example, the lock is mastered by node 1 (`owner=1`) and node 3 has been granted a protected-read lock on the file-system resource.

9. Run the following command, and look for processes that are in an uninterruptable sleep state as shown by the `D` flag in the `STAT` column.

    ```
    # ps -e -o pid,stat,comm,wchan=WIDE-WCHAN-COLUMN
    ```

    At least one of the processes that are in the uninterruptable sleep state will be responsible for the hang on the other node.

If a process is waiting for I/O to complete, the problem could be anywhere in the I/O subsystem from the block device layer through the drivers to the disk array. If the hang concerns a user lock (`flock()`), the problem could lie in the application. If possible, kill the holder of the lock. If the hang is due to lack of memory or fragmented memory, you can free up memory by killing non-essential processes. The most immediate solution is to reset the node that is holding the lock. The DLM recovery process can then clear all the locks that the dead node owned, so letting the cluster continue to operate.

23.3.5 Configuring the Behavior of Fenced Nodes

If a node with a mounted OCFS2 volume believes that it is no longer in contact with the other cluster nodes, it removes itself from the cluster in a process termed *fencing*. Fencing prevents other nodes from hanging when they try to access resources held by the fenced node. By default, a fenced node restarts instead of panicking so that it can quickly rejoin the cluster. Under some circumstances, you might want a fenced node to panic instead of restarting. For example, you might want to use `netconsole` to view the oops stack trace or to diagnose the cause of frequent reboots. To configure a node to panic when it next fences, run the following command on the node after the cluster starts:

```
# echo panic > /sys/kernel/config/cluster/cluster_name/fence_method
```

where `cluster_name` is the name of the cluster. To set the value after each reboot of the system, add this line to `/etc/rc.local`. To restore the default behavior, use the value `reset` instead of `panic`.

23.4 Use Cases for OCFS2

The following sections describe some typical use cases for OCFS2.

23.4.1 Load Balancing

You can use OCFS2 nodes to share resources between client systems. For example, the nodes could export a shared file system by using Samba or NFS. To distribute service requests between the nodes, you can use round-robin DNS, a network load balancer, or specify which node should be used on each client.

23.4.2 Oracle Real Application Cluster (RAC)

Oracle RAC uses its own cluster stack, Cluster Synchronization Services (CSS). You can use O2CB in conjunction with CSS, but you should note that each stack is configured independently for timeouts, nodes, and other cluster settings. You can use OCFS2 to host the voting disk files and the Oracle cluster registry (OCR), but not the grid infrastructure user's home, which must exist on a local file system on each node.

As both CSS and O2CB use the lowest node number as a tie breaker in quorum calculations, you should ensure that the node numbers are the same in both clusters. If necessary, edit the O2CB configuration file `/etc/ocfs2/cluster.conf` to make the node numbering consistent, and update this file on all nodes. The change takes effect when the cluster is restarted.

23.4.3 Oracle Databases

Specify the `noatime` option when mounting volumes that host Oracle datafiles, control files, redo logs, voting disk, and OCR. The `noatime` option disables unnecessary updates to the access time on the inodes.

Specify the `nointr` mount option to prevent signals interrupting I/O transactions that are in progress.

By default, the `init.ora` parameter `filesystemio_options` directs the database to perform direct I/O to the Oracle datafiles, control files, and redo logs. You should also specify the `datavolume` mount option for the volumes that contain the voting disk and OCR. Do not specify this option for volumes that host the Oracle user's home directory or Oracle E-Business Suite.

To avoid database blocks becoming fragmented across a disk, ensure that the file system cluster size is at least as big as the database block size, which is typically 8KB. If you specify the file system usage type as `datafiles` to the `mkfs.ocfs2` command, the file system cluster size is set to 128KB.

To allow multiple nodes to maximize throughput by concurrently streaming data to an Oracle datafile, OCFS2 deviates from the POSIX standard by not updating the modification time (`mtime`) on the disk when performing non-extending direct I/O writes. The value of `mtime` is updated in memory, but OCFS2 does not write the value to disk unless an application extends or truncates the file, or performs a operation to change the file metadata, such as using the `touch` command. This behavior leads to results in different nodes reporting different time stamps for the same file. You can use the following command to view the on-disk timestamp of a file:

```
# debugfs.ocfs2 -R "stat /file_path" device | grep "mtime:"
```

23.5 For More Information About OCFS2

You can find more information about OCFS2 at https://oss.oracle.com/projects/ocfs2/documentation/.

Part IV Authentication and Security

This section contains the following chapters:

- Chapter 24, *Authentication Configuration* describes how to configure various authentication methods that Oracle Linux can use, including NIS, LDAP, Kerberos, and Winbind, and how you can configure the System Security Services Daemon feature to provide centralized identity and authentication management.

- Chapter 25, *Local Account Configuration* describes how to configure and manage local user and group accounts.

- Chapter 26, *System Security Administration* describes the subsystems that you can use to administer system security, including SELinux, the Netfilter firewall, TCP Wrappers, chroot jails, auditing, system logging, and process accounting.

- Chapter 27, *OpenSSH Configuration* describes how to configure OpenSSH to support secure communication between networked systems.

Table of Contents

Chapter 24 Authentication Configuration

Table of Contents

This chapter describes how to configure various authentication methods that Oracle Linux can use, including NIS, LDAP, Kerberos, and Winbind, and how you can configure the System Security Services Daemon feature to provide centralized identity and authentication management.

24.1 About Authentication

Authentication is the verification of the identity of an entity, such as a user, to a system. A user logs in by providing a user name and a password, and the operating system authenticates the user's identity by comparing this information to data stored on the system. If the login credentials match and the user account is active, the user is authenticated and can successfully access the system.

The information that verifies a user's identity can either be located on the local system in the `/etc/passwd` and `/etc/shadow` files, or on remote systems using Identity Policy Audit (IPA), the Lightweight Directory Access Protocol (LDAP), the Network Information Service (NIS), or Winbind. In addition, IPSv2,

LDAP, and NIS data files can use the Kerberos authentication protocol, which allows nodes communicating over a non-secure network to prove their identity to one another in a secure manner.

You can use the Authentication Configuration GUI (`system-config-authentication`) to select the authentication mechanism and to configure any associated authentication options. Alternatively, you can use the `authconfig` command. Both the Authentication Configuration GUI and `authconfig` adjust settings in the PAM configuration files that are located in the `/etc/pam.d` directory. The Authentication Configuration GUI is available if you install the `authconfig-gtk` package.

Figure 24.1 shows the Authentication Configuration GUI with **Local accounts only** selected.

Figure 24.1 Authentication Configuration of Local Accounts
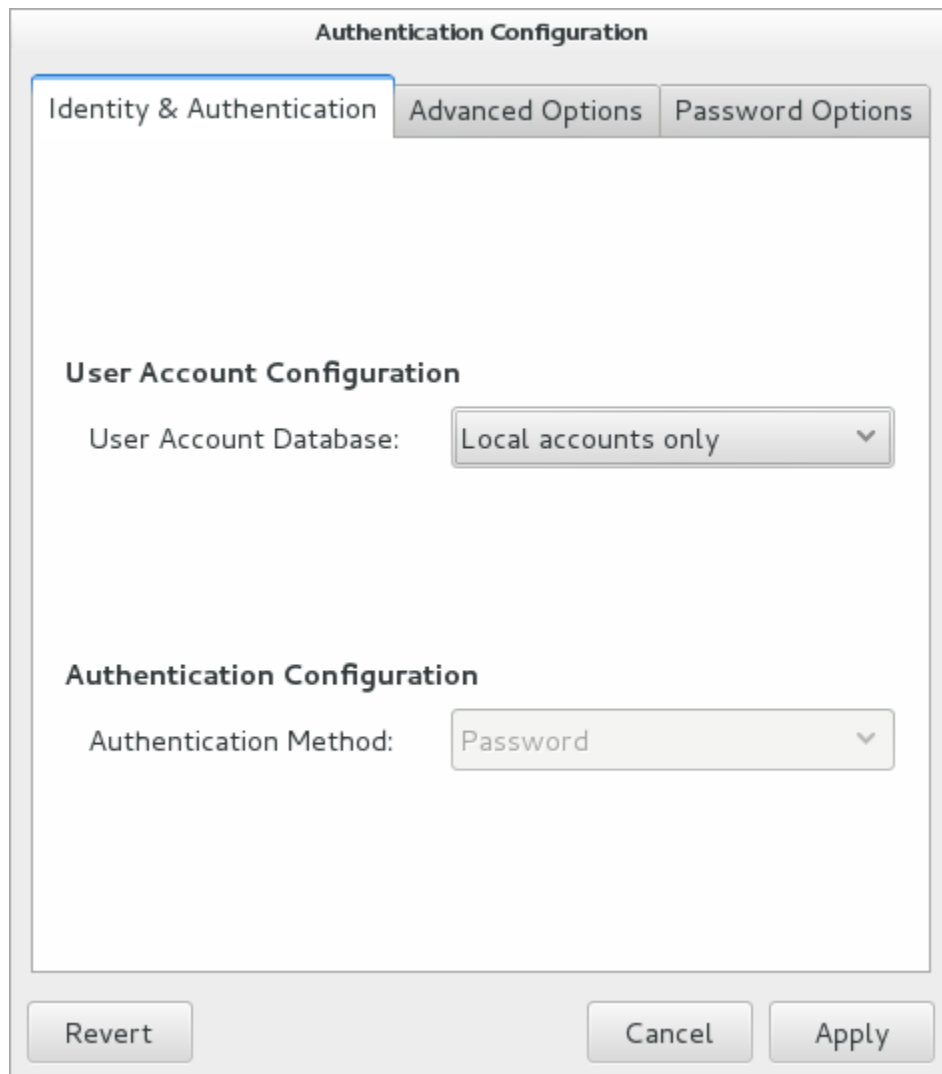
24.2 About Local Oracle Linux Authentication

Unless you select a different authentication mechanism during installation or by using the Authentication Configuration GUI or the `authconfig` command, Oracle Linux verifies a user's identity by using the information that is stored in the `/etc/passwd` and `/etc/shadow` files.

The `/etc/passwd` file stores account information for each user such as his or her unique user ID (or *UID*, which is an integer), user name, home directory, and login shell. A user logs in using his or her user name,

but the operating system uses the associated UID. When the user logs in, he or she is placed in his or her home directory and his or her login shell runs.

The `/etc/group` file stores information about groups of users. A user also belongs to one or more groups, and each group can contain one or more users. If you can grant access privileges to a group, all members of the group receive the same access privileges. Each group account has a unique group ID (*GID*, again an integer) and an associated group name.

By default, Oracle Linux implements the *user private group* (*UPG*) scheme where adding a user account also creates a corresponding UPG with the same name as the user, and of which the user is the only member.

Only the `root` user can add, modify, or delete user and group accounts. By default, both users and groups use shadow passwords, which are cryptographically hashed and stored in `/etc/shadow` and `/etc/gshadow` respectively. These shadow password files are readable only by the `root` user. root can set a group password that a user must enter to become a member of the group by using the `newgrp` command. If a group does not have a password, a user can only join the group by `root` adding him or her as a member.

The `/etc/login.defs` file defines parameters for password aging and related security policies.

For more information about the content of these files, see the `group(5)`, `gshadow(5)`, `login.defs(5)`, `passwd(5)`, and `shadow(5)` manual pages.

24.2.1 Configuring Local Access

You can use the User Manager GUI (`system-config-users`) to add or delete users and groups and to modify settings such as passwords, home directories, login shells, and group membership. Alternatively, you can use commands such as `useradd` and `groupadd`. The User Manager GUI is available if you install the `system-config-users` package.

To enable local access control, select the **Enable local access control** check box on the Advanced Options tab of the Authentication Configuration GUI (`system-config-authentication`). The system can then read the `/etc/security/access.conf` file for local user authorization rules that specify login combinations that the system accepts or refuses.

Figure 24.2 shows the Authentication Configuration GUI with the Advanced Options tab selected.

Figure 24.2 Authentication Configuration Advanced Options

Alternatively, use the following command:

```
# authconfig --enablepamaccess --update
```

Each entry in `/etc/security/access.conf` takes the form:

```
permission : users : origins [ except
```

where:

`permission`	Set to `+` or `–` to grant or deny login respectively.
`users`	Specifies a space-separated list of user or group names or `ALL` for any user or group. Enclose group names in parentheses to distinguish them from user names. You can use the `EXCEPT` operator to exclude a list of users from the rule.
`origins`	Specifies a space-separated list of host names, fully qualified domain names, network addresses, terminal device names, `ALL`, or `NONE`. You can use the `EXCEPT` operator to exclude a list of origins from the rule.

For example, the following rule denies login access by anyone except `root` from the network 192.168.2.0/24:

```
- : ALL except root : 192.168.2.0/24
```

For more information, see the `access.conf(5)` manual page and Chapter 25, *Local Account Configuration*.

24.2.2 Configuring Fingerprint Reader Authentication

If appropriate hardware is installed and supported, the system can use fingerprint scans to authenticate users.

To enable fingerprint reader support, select the **Enable fingerprint reader support** check box on the Advanced Options tab of the Authentication Configuration GUI (`system-config-authentication`).

Alternatively, use the following command:

```
# authconfig --enablefingerprint --update
```

24.2.3 Configuring Smart Card Authentication

If appropriate hardware is installed and supported, the system can use smart cards to authenticate users. The `pam_pkcs11` package provides a PAM login module that enables X.509 certificate-based user authentication. The module uses the Name Service Switch (NSS) to manage and validate PKCS #11 smart cards by using locally stored root CA certificates, online or locally accessible certificate revocation lists (CRLs), and the Online Certificate Status Protocol (OCSP).

To enable smart card authentication:

1. Install the `pam_pkcs11` package:

   ```
   # yum install pam_pkcs11
   ```

2. Use the following command to install the root CA certificates in the NSS database:

   ```
   # certutil -A -d /etc/pki/nssdb -t "TC,C,C" -n "Root CA certificates" -i CACert.pem
   ```

   where `CACert.pem` is the base-64 format root CA certificate file.

3. Run the Authentication Configuration GUI:

   ```
   # system-config-authentication
   ```

4. On the Advanced Options tab, select the **Enable smart card support** check box.

5. If you want to disable all other login authentication methods, select the **Require smart card for login** check box.

   > **Caution**
   >
   > Do not select this option until you have tested that can use a smart card to authenticate with the system.

6. From the **Card removal action** menu, select the system's response if a user removes a smart card while logged in to a session:

   Ignore   The system ignores card removal for the current session.

Lock The system locks the user out of the session .

You can also use the following command to configure smart card authentication:

```
# authconfig --enablesmartcard --update
```

To specify the system's response if a user removes a smart card while logged in to a session:

```
authconfig --smartcardaction=0|1 --update
```

Specify a value of 0 to `--smartcardaction` to lock the system if a card is removed. To ignore card removal, use a value of 1.

Once you have tested that you can use a smart card to authenticate with the system, you can disable all other login authentication methods.

```
# authconfig --enablerequiresmartcard --update
```

24.3 About IPA Authentication

IPA allows you to set up a domain controller for DNS, Kerberos, and authorization policies as an alternative to Active Directory Services. You can enrol client machines with an IPA domain so that they can access information for single sign-on authentication. IPA combines the capabilities of existing well-known technologies such as certificate services, DNS, LDAP, Kerberos, LDAP, and NTP.

24.3.1 Configuring IPA Authentication

To be able to configure IPA authentication, use `yum` to install the `ipa-client` and `ipa-admintools` packages. The `ipa-server` package is only required if you want to configure a system as an IPA server.

You can choose between two versions of IPA in the Authentication Configuration GUI:

* **FreeIPA** (effectively, IPAv1) supports identity management and authentication of users and groups, and does not require you to join your system to an IPA realm. Enter information about the LDAP and Kerberos configuration.

* **IPAv2**, which supports identity management and authentication of machines, requires you to join your system to an IPA realm. Enter information about the IPA domain configuration, optionally choose to configure NTP, and click **Join Domain** to create a machine account on the IPA server. After your system has obtained permission to join the IPA realm, you can select and configure the authentication method.

For more information about configuring IPA, see http://freeipa.org/page/Documentation.

24.4 About LDAP Authentication

The Lightweight Directory Access Protocol (LDAP) allows client systems to access information stored on LDAP servers over a network. An LDAP directory server stores information in a directory-based database that is optimized for searching and browsing, and which also supports simple functions for accessing and updating entries in the database.

Database entries are arranged in a hierarchical tree-like structure, where each directory can store information such as names, addresses, telephone numbers, network service information, printer information, and many other types of structured data. Systems can use LDAP for authentication, which allows users to access their accounts from any machine on a network.

The smallest unit of information in an LDAP directory is an entry, which can have one or more attributes. Each attribute of an entry has a name (also known as an *attribute type* or *attribute description*) and one

or more values. Examples of types are *domain component* (`dc`), *common name* (`cn`), organizational unit (`ou`) and email address (`mail`). The `objectClass` attribute allows you to specify whether an attribute is required or optional. An `objectClass` attribute's value specifies the schema rules that an entry must obey.

A *distinguished name* (`dn`) uniquely identifies an entry in LDAP. The distinguished name consists of the name of the entry (the *relative distinguished name* or RDN) concatenated with the names of its ancestor entries in the LDAP directory hierarchy. For example, the distinguished name of a user with the RDN `uid=arc815` might be `uid=arc815,ou=staff,dc=mydom,dc=com`.

The following are examples of information stored in LDAP for a user:

```
# User arc815
dn: uid=arc815,ou=People,dc=mydom,dc=com
cn: John Beck
givenName: John
sn: Beck
uid: arc815
uidNumber: 5159
gidNumber: 626
homeDirectory: /nethome/arc815
loginShell: /bin/bash
mail: johnb@mydom.com
objectClass: top
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
userPassword: {SSHA}QYrFtKkqOrifgk8H4EYf68B0JxIIaLga
```

and for a group:

```
# Group employees
dn: cn=employees,ou=Groups,dc=mydom,dc=com
cn: employees
gidNumber: 626
objectClass: top
objectClass: posixGroup
memberUid: arc815
memberUid: arc891
```

24.4.1 About LDAP Data Interchange Format

LDAP data itself is stored in a binary format. LDAP Data Interchange Format (LDIF) is a plain-text representation of an LDAP entry that allows the import and export LDAP data, usually to transfer the data between systems, but it also allows you to use a text editor to modify the content.

The data for an entry in an LDIF file takes the form:

```
[id] dn: distinguished_name
attribute_type: value | [attribute=]value[, [attribute=] value]...
...
objectClass: value
...
```

The optional `id` number is determined by the application that you use to edit the entry. Each attribute type for an entry contains either a value or a comma-separated list of attribute and value pairs as defined in the LDAP directory schema.

There must be a blank line between each `dn` definition section or `include:` line. There must not be any other blank lines or any white space at the ends of lines. White space at the start of a line indicates a continuation of the previous line.

24.4.2 Configuring an LDAP Server

OpenLDAP is an open-source implementation of LDAP that allows you configure an LDAP directory server.

To configure a system as an LDAP server:

1. Install the OpenLDAP packages:

    ```
    # yum install openldap openldap-servers openldap-clients nss-pam-ldapd
    ```

    The OpenLDAP configuration is stored in the following files below `/etc/openldap`:

`ldap.conf`	The configuration file for client applications.
`slapd.d/cn=config.ldif`	The default global configuration LDIF file for OpenLDAP.
`slapd.d/cn=config/*.ldif`	Configuration LDIF files for the database and schema.
`slapd.d/cn=config/cn=schema/*.ldif`	Schema configuration LDIF files. More information about the OpenLDAP schema is available at http://www.openldap.org/doc/admin/schema.html.

    Note

    You should never need to edit any files under `/etc/openldap/slapd.d` as you can reconfigure OpenLDAP while the `slapd` service is running.

2. If you want configure `slapd` to listen on port 636 for connections over an SSL tunnel (`ldaps://`), edit `/etc/sysconfig/slapd`, and change the value of `SLAPD_LDAPS` to `yes`:

    ```
    SLAPD_LDAPS=yes
    ```

    If required, you can prevent `slapd` listening on port 389 for `ldap://` connections, by changing the value of `SLAPD_LDAP` to `no`:

    ```
    SLAPD_LDAP=no
    ```

    Ensure that you also define the correct SLAPD_URLS for the ports that are enabled. For instance, if you intend to use SSL and you wish `slapd` to listen on port 636, you must specify `ldaps://` as one of the supported URLS. For example:

    ```
    SLAPD_URLS="ldapi:/// ldap:/// ldaps:///"
    ```

3. Configure the system firewall to allow incoming TCP connections on port 389, for example:

    ```
    # firewall-cmd --zone=zone --add-port=389/tcp
    # firewall-cmd --permanent --zone=zone --add-port=389/tcp
    ```

    The primary TCP port for LDAP is 389. If you configure LDAP to use an SSL tunnel (`ldaps`), substitute the port number that the tunnel uses, which is usually 636, for example:

    ```
    # firewall-cmd --zone=zone --add-port=636/tcp
    # firewall-cmd --permanent --zone=zone --add-port=636/tcp
    ```

4. Change the user and group ownership of `/var/lib/ldap` and any files that it contains to `ldap`:

    ```
    # cd /var/lib/ldap
    # chown ldap:ldap ./*
    ```

5. Start the `slapd` service and configure it to start following system reboots:

```
# systemctl start slapd
# systemctl enable slapd
```

6. Generate a hash of the LDAP password that you will use with the `olcRootPW` entry in the configuration file for your domain database, for example:

```
# slappasswd -h {SSHA}
New password: password
Re-enter new password: password
{SSHA}lkMShz73MZBic19Q4pfOaXNxpLN3wLRy
```

7. Create an LDIF file with a name such as `config-mydom-com.ldif` that contains configuration entries for your domain database based on the following example:

```
# Load the schema files required for accounts
include file:///etc/openldap/schema/cosine.ldif

include file:///etc/openldap/schema/nis.ldif

include file:///etc/openldap/schema/inetorgperson.ldif

# Load the HDB (hierarchical database) backend modules
dn: cn=module,cn=config
objectClass: olcModuleList
cn: module
olcModulepath: /usr/lib64/openldap
olcModuleload: back_hdb

# Configure the database settings
dn: olcDatabase=hdb,cn=config
objectClass: olcDatabaseConfig
objectClass: olcHdbConfig
olcDatabase: {1}hdb
olcSuffix: dc=mydom,dc=com
# The database directory must already exist
# and it should only be owned by ldap:ldap.
# Setting its mode to 0700 is recommended
olcDbDirectory: /var/lib/ldap
olcRootDN: cn=admin,dc=mydom,dc=com
olcRootPW: {SSHA}lkMShz73MZBic19Q4pfOaXNxpLN3wLRy
olcDbConfig: set_cachesize 0 10485760 0
olcDbConfig: set_lk_max_objects 2000
olcDbConfig: set_lk_max_locks 2000
olcDbConfig: set_lk_max_lockers 2000
olcDbIndex: objectClass eq
olcLastMod: TRUE
olcDbCheckpoint: 1024 10
# Set up access control
olcAccess: to attrs=userPassword
  by dn="cn=admin,dc=mydom,dc=com"
  write by anonymous auth
  by self write
  by * none
olcAccess: to attrs=shadowLastChange
  by self write
  by * read
olcAccess: to dn.base=""
  by * read
olcAccess: to *
  by dn="cn=admin,dc=mydom,dc=com"
  write by * read
```

> **Note**
>
> This configuration file allows you to reconfigure `slapd` while it is running. If you use a `slapd.conf` configuration file, you can also update `slapd` dynamically, but such changes do not persist if you restart the server.

For more information, see the `slapd-config(5)` manual page.

8. Use the `ldapadd` command to add the LDIF file:

```
# ldapadd -Y EXTERNAL -H ldapi:/// -f config-mydom-com.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
adding new entry "cn=module,cn=config"

adding new entry "olcDatabase=hdb,cn=config"
```

For more information about configuring OpenLDAP, see the `slapadd(8C)`, `slapd(8C)`, `slapd-config(5)`, and `slappasswd(8C)` manual pages, the *OpenLDAP Administrator's Guide* (`/usr/share/doc/openldap-servers-version/guide.html`), and the latest OpenLDAP documentation at http://www.openldap.org/doc/.

24.4.3 Replacing the Default Certificates

If you configure LDAP to use Transport Layer Security (TLS) or Secure Sockets Layer (SSL) to secure the connection to the LDAP server, you need a public certificate that clients can download. You can obtain certificates from a Certification Authority (CA) or you can use the `openssl` command to create the certificate. See Section 24.4.4, "Creating and Distributing Self-signed CA Certificates".

Once you have a server certificate, its corresponding private key file, and a root CA certificate, you can replace the default certificates that are installed in `/etc/openldap/certs`.

To display the existing certificate entries that `slapd` uses with TLS, use the `ldapsearch` command:

```
# ldapsearch -LLL -Y EXTERNAL -H ldapi:/// -b "cn=config" \
  olcTLSCACertificatePath olcTLSCertificateFile olcTLSCertificateKeyFile
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
dn: cn=config
olcTLSCACertificatePath: /etc/openldap/certs
olcTLSCertificateFile: "OpenLDAP Server"
olcTLSCertificateKeyFile: /etc/openldap/certs/password
...
```

To replace the TLS attributes in the LDAP configuration:

1. Create an LDIF file that defines how to modify the attributes, for example:

```
dn: cn=config
changetype: modify
delete: olcTLSCACertificatePath

# Omit the following clause for olcTLSCACertificateFile
# if you do not have a separate root CA certificate
dn: cn=config
changetype: modify
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/certsCAcert.pem

dn: cn=config
```

```
changetype: modify
replace: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ssl/certs/server-cert.pem

dn: cn=config
changetype: modify
replace: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ssl/certs/server-key.pem

dn: cn=config
changetype: modify
add: olcTLSCipherSuite
olcTLSCipherSuite: TLSv1+RSA:!NULL

dn: cn=config
changetype: modify
add: olcTLSVerifyClient
olcTLSVerifyClient: never
```

If you generate only a self-signed certificate and its corresponding key file, you do not need to specify a root CA certificate.

2. Use the `ldapmodify` command to apply the LDIF file:

```
# ldapmodify -Y EXTERNAL -H ldapi:/// -f mod-TLS.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=config"

modifying entry "cn=config"

modifying entry "cn=config"
...
```

3. Verify that the entries have changed:

```
# ldapsearch -LLL -Y EXTERNAL -H ldapi:/// -b "cn=config" \
  olcTLSCACertificatePath olcTLSCertificateFile olcTLSCertificateKeyFile
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
dn: cn=config
olcTLSCACertificateFile: /etc/ssl/certs/CAcert.pem
olcTLSCertificateFile: /etc/ssl/certs/server-cert.pem
olcTLSCertificateKeyFile: /etc/ssl/certs/server-key.pem
olcTLSCipherSuite: TLSv1+RSA:!NULL
olcTLSVerifyClient: never
...
```

4. Restart the `slapd` service to make it use the new certificates:

```
# systemctl restart slapd
```

For more information, see the `ldapmodify(1)`, `ldapsearch(1)` and `openssl(1)` manual pages.

24.4.4 Creating and Distributing Self-signed CA Certificates

For usage solely within an organization, you might want to create certificates that you can use with LDAP. There are a number of ways of creating suitable certificates, for example:

- Create a self-signed CA certificate together with a private key file.

- Create a self-signed root CA certificate and private key file, and use the CA certificate and its key file to sign a separate server certificate for each server.

The following procedure describes how to use `openssl` to create a self-signed CA certificate and private key file, and then use these files to sign server certificates.

To create the CA certificate and use it to sign a server certificate:

1. Change directory to `/etc/openldap/certs` on the LDAP server:

   ```
   # cd /etc/openldap/certs
   ```

2. Create the private key file `CAcert-key.pem` for the CA certificate:

   ```
   # openssl genrsa -out CAcert-key.pem 1024
   Generating RSA private key, 1024 bit long modulus
   ......++++++
   ....++++++
   e is 65537 (0x10001)
   ```

3. Change the mode on the key file to 0400:

   ```
   # chmod 0400 CAcert-key.pem
   ```

4. Create the certificate request `CAcert.csr`:

   ```
   # openssl req -new -key CAcert-key.pem -out CAcert.csr
   You are about to be asked to enter information that will be incorporated
   into your certificate request.
   What you are about to enter is what is called a Distinguished Name or a DN.
   There are quite a few fields but you can leave some blank
   For some fields there will be a default value,
   If you enter '.', the field will be left blank.
   -----
   Country Name (2 letter code) [XX]:US
   State or Province Name (full name) []:California
   Locality Name (eg, city) [Default City]:Redwood City
   Organization Name (eg, company) [Default Company Ltd]:Mydom Inc
   Organizational Unit Name (eg, section) []:Org
   Common Name (eg, your name or your server's hostname) []:www.mydom.org
   Email Address []:root@mydom.org

   Please enter the following 'extra' attributes
   to be sent with your certificate request
   A challenge password []:<Enter>
   An optional company name []:<Enter>
   ```

5. Create a CA certificate that is valid for approximately three years:

   ```
   # openssl x509 -req -days 1095 -in CAcert.csr -signkey CAcert-key.pem -out CAcert.pem
   rt-key.pem -out CAcert.pem
   Signature ok
   subject=/C=US/ST=California/L=Redwood City/O=Mydom
   Inc/OU=Org/CN=www.mydom.org/emailAddress=root@mydom.org
   Getting Private key
   ```

6. For each server certificate that you want to create:

   a. Create the private key for the server certificate:

   ```
   # openssl genrsa -out server-key.pem 1024
   Generating RSA private key, 1024 bit long modulus
   .............++++++
   ...........................++++++
   e is 65537 (0x10001)
   ```

Note

If you intend to generate server certificates for several servers, name the certificate, its key file, and the certificate request so that you can easily identify both the server and the service, for example, `ldap_host02-cert.pem`, `ldap_host02-key.pem`, and `ldap_host02-cert.csr`.

b. Change the mode on the key file to 0400, and change its user and group ownership to `ldap`:

```
# chmod 0400 server-key.pem
# chown ldap:ldap server-key.pem
```

c. Create the certificate request `server-cert.csr`:

```
# openssl req -new -key server-key.pem -out server-cert.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:California
Locality Name (eg, city) [Default City]:Redwood City
Organization Name (eg, company) [Default Company Ltd]:Mydom Inc
Organizational Unit Name (eg, section) []:Org
Common Name (eg, your name or your server's hostname) []:ldap.mydom.com
Email Address []:root@mydom.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:<Enter>
An optional company name []:<Enter>
```

Note

For the Common Name, specify the Fully Qualified Domain Name (FQDN) of the server. If the FQDN of the server does not match the common name specified in the certificate, clients cannot obtain a connection to the server.

d. Use the CA certificate and its corresponding key file to sign the certificate request and generate the server certificate:

```
# openssl x509 -req -days 1095 -CAcreateserial \
  -in server-cert.csr -CA CAcert.pem -CAkey CAcert-key.pem \
  -out server-cert.pem
Signature ok
subject=/C=US/ST=California/L=Redwood City/O=Mydom
Inc/OU=Org/CN=ldap.mydom.com/emailAddress=root@mydom.com
Getting CA Private Key
```

7. If you generate server certificates for other LDAP servers, copy the appropriate server certificate, its corresponding key file, and the CA certificate to `/etc/openldap/certs` on those servers.

8. Set up a web server to host the CA certificate for access by clients. The following steps assume that the LDAP server performs this function. You can use any suitable, alternative server instead.

a. Install the Apache HTTP server.

```
# yum install httpd
```

b. Create a directory for the CA certificate under `/var/www/html`, for example:

```
# mkdir /var/www/html/certs
```

c. Copy the CA certificate to `/var/www/html/certs`.

```
# cp CAcert.pem /var/www/html/certs
```

Caution

Do not copy the key files.

d. Edit the HTTP server configuration file, `/etc/httpd/conf/httpd.conf`, and specify the resolvable domain name of the server in the argument to `ServerName`.

```
ServerName server_addr:80
```

If the server does not have a resolvable domain name, enter its IP address instead.

Verify that the setting of the `Options` directive in the `<Directory "/var/www/html">` section specifies `Indexes` and `FollowSymLinks` to allow you to browse the directory hierarchy, for example:

```
Options Indexes FollowSymLinks
```

e. Start the Apache HTTP server, and configure it to start after a reboot.

```
# systemctl start httpd
# systemctl enable httpd
```

f. If you have enabled the firewall on your system, configure it to allow incoming HTTP connection requests on TCP port 80, for example:

```
# firewall-cmd --zone=zone --add-port=80/tcp
# firewall-cmd --permanent --zone=zone --add-port=80/tcp
```

24.4.5 Initializing an Organization in LDAP

Before you can define people, groups, servers, printers, and other entitles for your organization, you must first set up information in LDAP for the organization itself.

To define an organization in LDAP:

1. Create an LDIF file that defines the organization, for example `mydom-com-organization.ldif`:

```
# Organization mydom.com
dn: dc=mydom,dc=com
dc: mydom
objectclass: dcObject
objectclass: organizationalUnit
ou: mydom.com

# Users
dn: ou=People,dc=mydom,dc=com
objectClass: organizationalUnit
ou: people

# Groups
dn: ou=Groups,dc=mydom,dc=com
objectClass: organizationalUnit
```

```
ou: groups
```

2. If you have configured LDAP authentication, use the `ldapadd` command to add the organization to LDAP:

```
# ldapadd -cxWD "cn=admin,dc=mydom,dc=com" -f mydom-com-organization.ldif
Enter LDAP Password: admin_password
adding new entry "dc=mydom,dc=com"

adding new entry "ou=People,dc=mydom,dc=com"

adding new entry "ou=Groups,dc=mydom,dc=com"
```

If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the `ldapadd` command:

```
# ldapadd -f mydom-com-organization.ldif
```

For more information, see the `ldapadd(1)` manual page.

24.4.6 Adding an Automount Map to LDAP

You can make an automount map such as `auto.home` available in LDAP so that the automounter mounts a user's home directory on demand.

To add the `auto.home` map to LDAP:

1. Create an LDIF file that defines entries for the map's name and its contents, for example `auto-home.ldif`:

```
dn: nisMapName=auto.home,dc=mydom,dc=com
objectClass: top
objectClass: nisMap
nisMapName: auto.home

dn: cn=*,nisMapName=auto.home,dc=mydom,dc=com
objectClass: nisObject
cn: *
nisMapEntry: -rw,sync nfssvr:/nethome/&
nisMapName: auto.home
```

where `nfssvr` is the host name or IP address of the NFS server that exports the users' home directories.

2. If you have configured LDAP authentication, use the following command to add the map to LDAP:

```
# ldapadd -xcWD "cn=admin,dc=mydom,dc=com" \
  -f auto-home.ldif
Enter LDAP Password: user_password
adding new entry "nisMapName=auto.home,dc=mydom,dc=com"

adding new entry "cn=*,nisMapName=auto.home,dc=mydom,dc=com"
```

If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the command:

```
# ldapmodify -f auto-home.ldif
```

3. Verify that the map appears in LDAP:

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" nisMapName=auto.home
dn: nisMapName=auto.home,dc=mydom,dc=com
```

```
objectClass: top
objectClass: nisMap
nisMapName: auto.home

dn: cn=*,nisMapName=auto.home,dc=mydom,dc=com
objectClass: nisObject
cn: *
nisMapEntry: -rw,sync nfssvr.mydom.com:/nethome/&
nisMapName: auto.home
```

24.4.7 Adding a Group to LDAP

If you configure users in user private groups (UPGs), define that group along with the user. See Section 24.4.8, "Adding a User to LDAP".

To add a group to LDAP:

1. Create an LDIF file that defines the group, for example `employees-group.ldif`:

   ```
   # Group employees
   dn: cn=employees,ou=Groups,dc=mydom,dc=com
   cn: employees
   gidNumber: 626
   objectClass: top
   objectclass: posixGroup
   ```

2. If you have configured LDAP authentication, use the following command to add the group to LDAP:

   ```
   # ldapadd -cxWD "cn=admin,dc=mydom,dc=com" -f employees-group.ldif
   Enter LDAP Password: admin_password
   adding new entry "cn=employees,ou=Groups,dc=mydom,dc=com"
   ```

   If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the `ldapadd` command:

   ```
   # ldapadd -f employees-group.ldif
   ```

3. Verify that you can locate the group in LDAP:

   ```
   # ldapsearch -LLL -x -b "dc=mydom,dc=com" gidNumber=626
   dn: cn=employees,ou=Groups,dc=mydom,dc=com
   cn: employees
   gidNumber: 626
   objectClass: top
   objectClass: posixGroup
   ```

For more information, see the `ldapadd(1)` and `ldapsearch(1)` manual pages.

24.4.8 Adding a User to LDAP

Note

This procedure assumes that:

- LDAP provides information for `ou=People`, `ou=Groups`, and `nisMapName=auto.home`.

- The LDAP server uses NFS to export the users' home directories. See Section 22.2.2, "Mounting an NFS File System"

To create an account for a user on the LDAP server:

1. If the LDAP server does not already export the base directory of the users' home directories, perform the following steps on the LDAP server:

   a. Create the base directory for user directories, for example `/nethome`:

   ```
   # mkdir /nethome
   ```

   b. Add an entry such as the following to `/etc/exports`:

   ```
   /nethome      *(rw,sync)
   ```

   You might prefer to restrict which clients can mount the file system. For example, the following entry allows only clients in the 192.168.1.0/24 subnet to mount `/nethome`:

   ```
   /nethome      192.168.1.0/24(rw,sync)
   ```

   c. Use the following command to export the file system:

   ```
   # exportfs -i -o ro,sync *:/nethome
   ```

2. Create the user account, but do not allow local logins:

   ```
   # useradd -b base_dir -s /sbin/nologin -u UID -U username
   ```

   For example:

   ```
   # useradd -b /nethome -s /sbin/nologin -u 5159 -U arc815
   ```

   The command updates the `/etc/passwd` file and creates a home directory under `/nethome` on the LDAP server.

   The user's login shell will be overridden by the `LoginShell` value set in LDAP.

3. Use the `id` command to list the user and group IDs that have been assigned to the user, for example:

   ```
   # id arc815
   uid=5159(arc815) gid=5159(arc815) groups=5159(arc815)
   ```

4. Create an LDIF file that defines the user, for example `arc815-user.ldif`:

   ```
   # UPG arc815
   dn: cn=arc815,ou=Groups,dc=mydom,dc=com
   cn: arc815
   gidNumber: 5159
   objectclass: top
   objectclass: posixGroup

   # User arc815
   dn: uid=arc815,ou=People,dc=mydom,dc=com
   cn: John Beck
   givenName: John
   sn: Beck
   uid: arc815
   uidNumber: 5159
   gidNumber: 5159
   homeDirectory: /nethome/arc815
   loginShell: /bin/bash
   mail: johnb@mydom.com
   objectClass: top
   objectClass: inetOrgPerson
   objectClass: posixAccount
   objectClass: shadowAccount
   userPassword: {SSHA}x
   ```

In this example, the user belongs to a user private group (UPG), which is defined in the same file. The user's login shell attribute `LoginShell` is set to `/bin/bash`. The user's password attribute `userPassword` is set to a placeholder value. If you use Kerberos authentication with LDAP, this attribute is not used.

5. If you have configured LDAP authentication, use the following command to add the user to LDAP:

```
# ldapadd -cxWD cn=admin,dc=mydom,dc=com -f arc815-user.ldif
Enter LDAP Password: admin_password
adding new entry "cn=arc815,ou=Groups,dc=mydom,dc=com"

adding new entry "uid=arc815,ou=People,dc=mydom,dc=com"
```

If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the `ldapadd` command:

```
# ldapadd -f arc815-user.ldif
```

6. Verify that you can locate the user and his or her UPG in LDAP:

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" '(|(uid=arc815)(cn=arc815))'
dn: cn=arc815,ou=Groups,dc=mydom,dc=com
cn: arc815
gidNumber: 5159
objectClass: top
objectClass: posixGroup

dn: uid=arc815,ou=People,dc=mydom,dc=com
cn: John Beck
givenName: John
sn: Beck
uid: arc815
uidNumber: 5159
gidNumber: 5159
homeDirectory: /home/arc815
loginShell: /bin/bash
mail: johnb@mydom.com
objectClass: top
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
```

7. If you have configured LDAP authentication, set the user password in LDAP:

```
# ldappasswd -xWD "cn=admin,dc=mydom,dc=com" \
  -S "uid=arc815,ou=people,dc=mydom,dc=com"
New password: user_password
Re-enter new password: user_password
Enter LDAP Password: admin_password
```

If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use the `kadmin` command to add the user (principal) and password to the database for the Kerberos domain, for example:

```
# kadmin -q "addprinc alice@MYDOM.COM"
```

For more information, see the `kadmin(1)`, `ldapadd(1)`, `ldappasswd(1)`, and `ldapsearch(1)` manual pages.

24.4.9 Adding Users to a Group in LDAP

To add users to an existing group in LDAP:

1. Create an LDIF file that defines the users that should be added to the `memberuid` attribute for the group, for example `employees-add-users.ldif`:

```
dn: cn=employees,ou=Groups,dc=mydom,dc=com
changetype: modify
add: memberUid
memberUid: arc815

dn: cn=employees,ou=Groups,dc=mydom,dc=com
changetype: modify
add: memberUid
memberUid: arc891


...
```

2. If you have configured LDAP authentication, use the following command to add the group to LDAP:

```
# ldapmodify -xcWD "cn=admin,dc=mydom,dc=com" \
  -f employees-add-users.ldif
Enter LDAP Password: user_password
modifying entry "cn=employees,ou=Groups,dc=mydom,dc=com"
...
```

If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the command:

```
# ldapmodify -f employees-add-users.ldif
```

3. Verify that the group has been updated in LDAP:

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" gidNumber=626
dn: cn=employees,ou=Groups,dc=mydom,dc=com
cn: employees
gidNumber: 626
objectClass: top
objectClass: posixGroup
memberUid: arc815
memberUid: arc891
...
```

24.4.10 Enabling LDAP Authentication

To enable LDAP authentication for an LDAP client by using the Authentication Configuration GUI:

1. Install the `openldap-clients` package:

```
# yum install openldap-clients
```

2. Run the Authentication Configuration GUI:

```
# system-config-authentication
```

3. Select **LDAP** as the user account database and enter values for:

LDAP Search Base DN	The LDAP Search Base DN for the database. For example: `dc=mydom,dc=com`.
LDAP Server	The URL of the LDAP server including the port number. For example, `ldap://ldap.mydom.com:389` or `ldaps://ldap.mydom.com:636`.

LDAP authentication requires that you use either LDAP over SSL (`ldaps`) or Transport Layer Security (TLS) to secure the connection to the LDAP server.

4. If you use TLS, click **Download CA Certificate** and enter the URL from which to download the CA certificate that provides the basis for authentication within the domain.

5. Select either **LDAP password** or **Kerberos password** for authentication.

6. If you select Kerberos authentication, enter values for:

Realm	The name of the Kerberos realm.
KDCs	A comma-separated list of Key Distribution Center (KDC) servers that can issue Kerberos ticket granting tickets and service tickets.
Admin Servers	A comma-separated list of Kerberos administration servers.

Alternatively, you can use DNS to configure these settings:

- Select the **Use DNS to resolve hosts to realms** check box to look up the name of the realm defined as a `TXT` record in DNS, for example:

```
_kerberos.mydom.com      IN TXT "MYDOM.COM"
```

- Select the **Use DNS to locate KDCs for realms** check box to look up the KDCs and administration servers defined as `SVR` records in DNS, for example:

```
_kerberos._tcp.mydom.com      IN SVR 1  0 88   krbsvr.mydom.com
_kerberos._udp.mydom.com      IN SVR 1  0 88   krbsvr.mydom.com
_kpasswd._udp.mydom.com       IN SVR 1  0 464 krbsvr.mydom.com
_kerberos-adm._tcp.mydom.com  IN SVR 1  0 749 krbsvr.mydom.com
```

7. Click **Apply** to save your changes.

Figure 24.3 shows the Authentication Configuration GUI with LDAP selected for the user account database and for authentication.

Figure 24.3 Authentication Configuration Using LDAP

You can also enable LDAP by using the `authconfig` command.

To use LDAP as the authentication source, specify the `--enableldapauth` option together with the full LDAP server URL including the port number and the LDAP Search Base DN, as shown in the following example:.

```
# authconfig --enableldap --enableldapauth \
  --ldapserver=ldaps://ldap.mydom.com:636 \
  --ldapbasedn="ou=people,dc=mydom,dc=com" \
  --update
```

If you want to use TLS, additionally specify the `--enableldaptls` option and the download URL of the CA certificate, for example:

```
# authconfig --enableldap --enableldapauth \
  --ldapserver=ldap://ldap.mydom.com:389 \
  --ldapbasedn="ou=people,dc=mydom,dc=com" \
  --enableldaptls \
  --ldaploadcacert=https://ca-server.mydom.com/CAcert.pem \
  --update
```

The `--enableldap` option configures `/etc/nsswitch.conf` to enable the system to use LDAP and SSSD for information services. The `--enableldapauth` option enables LDAP authentication by modifying the PAM configuration files in `/etc/pam.d` to use the `pam_ldap.so` module.

For more information, see the `authconfig(8)`, `pam_ldap(5)`, and `nsswitch.conf(5)` manual pages.

For information about using Kerberos authentication with LDAP, see Section 24.6.3, "Enabling Kerberos Authentication".

> **Note**
>
> You must also configure SSSD to be able to access information in LDAP. See Section 24.4.10.1, "Configuring an LDAP Client to use SSSD".
>
> If your client uses automount maps stored in LDAP, you must configure `autofs` to work with LDAP. See Section 24.4.10.2, "Configuring an LDAP Client to Use Automount Maps".

24.4.10.1 Configuring an LDAP Client to use SSSD

The Authentication Configuration GUI and `authconfig` configure access to LDAP via `sss` entries in `/etc/nsswitch.conf` so you must configure the System Security Services Daemon (SSSD) on the LDAP client.

To configure an LDAP client to use SSSD:

1. Install the `sssd` and `sssd-client` packages:

   ```
   # yum install sssd sssd-client
   ```

2. Edit the `/etc/sssd/sssd.conf` configuration file and configure the sections to support the required services, for example:

   ```
   [sssd]
   config_file_version = 2
   domains = default
   services = nss, pam

   [domain/default]
   id_provider = ldap
   ldap_uri = ldap://ldap.mydom.com
   ldap_id_use_start_tls = true
   ldap_search_base = dc=mydom,dc=com
   ldap_tls_cacertdir = /etc/openldap/cacerts

   auth_provider = krb5
   chpass_provider = krb5
   krb5_realm = MYDOM.COM
   krb5_server = krbsvr.mydom.com
   krb5_kpasswd = krbsvr.mydom.com
   cache_credentials = true

   [domain/LDAP]
   id_provider = ldap
   ldap_uri = ldap://ldap.mydom.com
   ldap_search_base = dc=mydom,dc=com

   auth_provider = krb5
   krb5_realm = MYDOM.COM
   krb5_server = kdcsvr.mydom.com
   cache_credentials = true

   min_id = 5000
   ```

```
max_id = 25000
enumerate = false

[nss]
filter_groups = root
filter_users = root
reconnection_retries = 3
entry_cache_timeout = 300

[pam]
reconnection_retries = 3
offline_credentials_expiration = 2
offline_failed_login_attempts = 3
offline_failed_login_delay = 5
```

3. Change the mode of `/etc/sssd/sssd.conf` to 0600:

```
# chmod 0600 /etc/sssd/sssd.conf
```

4. Enable the SSSD service:

```
# authconfig --update --enablesssd --enablesssdauth
```

For more information, see the `sssd.conf(5)` manual page and Section 24.8, "About the System Security Services Daemon".

24.4.10.2 Configuring an LDAP Client to Use Automount Maps

If you have configured an automount map for `auto.home` in LDAP, you can configure an LDAP client to mount the users' home directories when they log in.

To configure an LDAP client to automount users' home directories:

1. Install the `autofs` package:

```
# yum install autofs
```

2. Verify that the `auto.home` map is available :

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" nisMapName=auto.home
dn: nisMapName=auto.home,dc=mydom,dc=com
objectClass: top
objectClass: nisMap
nisMapName: auto.home

dn: cn=*,nisMapName=auto.home,dc=mydom,dc=com
objectClass: nisObject
cn: *
nisMapEntry: -rw,sync nfssvr.mydom.com:/nethome/&
nisMapName: auto.home
```

In this example, the map is available. For details of how to make this map available, see Section 24.4.6, "Adding an Automount Map to LDAP".

3. If the `auto.home` map is available, edit `/etc/auto.master` and create an entry that tells `autofs` where to find the `auto.home` map in LDAP, for example:

```
/nethome      ldap:nisMapName=auto.home,dc=mydom,dc=com
```

If you use LDAP over SSL, specify `ldaps:` instead of `ldap:`.

4. Edit `/etc/autofs_ldap_auth.conf` and configure the authentication settings for `autofs` with LDAP, for example:

```
<autofs_ldap_sasl_conf
     usetls="yes"
     tlsrequired="no"
     authrequired="autodetect"
     authtype="GSSAPI"
     clientprinc="host/ldapclient.mydom.com@MYDOM.COM"
     />
```

This example assumes that Kerberos authentication with the LDAP server uses TLS for the connection. The principal for the client system must exist in the Kerberos database. You can use the `klist -k` command to verify this. If the principal for the client does not exist, use `kadmin` to add the principal.

5. If you use Kerberos Authentication, use `kadmin` to add a principal for the LDAP service on the LDAP server, for example:

```
# kadmin -q "addprinc ldap/ldap.mydom.com@MYDOM.COM
```

6. Restart the `autofs` service, and configure the service to start following a system reboot:

```
# systemctl restart autofs
# systemctl enable autofs
```

The `autofs` service creates the directory `/nethome`. When a user logs in, the automounter mounts his or her home directory under `/nethome`.

If the owner and group for the user's files are unexpectedly listed as the anonymous user or group (`nobody` or `nogroup`) and `all_squash` has not been specified as a mount option, verify that the `Domain` setting in `/etc/idmapd.conf` on the NFS server is set to the DNS domain name. Restart the NFS services on the NFS server if you change this file.

For more information, see the `auto.master(5)` and `autofs_ldap_auth.conf(5)` manual pages.

24.5 About NIS Authentication

NIS stores administrative information such as user names, passwords, and host names on a centralized server. Client systems on the network can access this common data. This configuration allows to move from machine to machine without having to remember different passwords and copy data from one machine to another. Storing administrative information centrally, and providing a means of accessing it from networked systems, also ensures the consistency of that data. NIS also reduces the overhead of maintaining administration files such as `/etc/passwd` on each system.

A network of NIS systems is an *NIS domain*. Each system within the domain has the same NIS domain name, which is different from a DNS domain name. The DNS domain is used throughout the Internet to refer to a group of systems. an NIS domain is used to identify systems that use files on an NIS server. an NIS domain must have exactly one master server but can have multiple slave servers.

24.5.1 About NIS Maps

The administrative files within an NIS domain are NIS maps, which are `dbm`-format files that you generate from existing configuration files such as `/etc/passwd`, `/etc/shadow`, and `/etc/groups`. Each map is indexed on one field, and records are retrieved by specifying a value from that field. Some source files such as `/etc/passwd` have two maps:

`passwd.byname`	Indexed on user name.
`passwd.byuid`	Indexed on user ID.

The `/var/yp/nicknames` file contains a list of commonly used short names for maps such as `passwd` for `passwd.byname` and `group` for `group.byname`.

You can use the `ypcat` command to display the contents of an NIS map, for example:

```
# ypcat - passwd | grep 1500
guest:$6$gMIxsr3W$LaAo...6EE6sdsFPI2mdm7/NEm0:1500:1500::/nethome/guest:/bin/bash
```

> **Note**
>
> As the `ypcat` command displays password hashes to any user, this example demonstrates that NIS authentication is inherently insecure against password-hash cracking programs. If you use Kerberos authentication, you can configure password hashes not to appear in NIS maps, although other information that `ypcat` displays could also be useful to an attacker.

For more information, see the `ypcat(1)` manual page.

24.5.2 Configuring an NIS Server

NIS master servers act as a central, authoritative repository for NIS information. NIS slave servers act as mirrors of this information. There must be only one NIS master server in an NIS domain. The number of NIS slave servers is optional, but creating at least one slave server provides a degree of redundancy should the master server be unavailable.

To configure an NIS master or slave server:

1. Install the `ypserv` package:

   ```
   # yum install ypserv
   ```

2. Edit `/etc/sysconfig/network` and add an entry to define the NIS domain, for example:

   ```
   NISDOMAIN=mynisdom
   ```

3. Edit `/etc/ypserv.conf` to configure NIS options and to add rules for which hosts and domains can access which NIS maps.

   For example, the following entries allow access only to NIS clients in the `mynisdom` domain on the 192.168.1 subnet:

   ```
   192.168.1.0/24: mynisdom : * : none
   * : * : * : deny
   ```

   For more information, see the `ypserv.conf(5)` manual page and the comments in `/etc/ypserv.conf`.

4. Create the file `/var/yp/securenets` and add entries for the networks for which the server should respond to requests, for example:

   ```
   # cat > /var/yp/securenets <<!
   255.255.255.255 127.0.0.1
   255.255.255.0   192.168.1.0
   !
   # cat /var/yp/securenets
   255.255.255.255 127.0.0.1
   255.255.255.0   192.168.1.0
   ```

   In this example, the server accepts requests from the local loopback interface and the 192.168.1 subnet.

5. Edit `/var/yp/Makefile`:

    a. Set any required map options and specify which NIS maps to create using the `all` target, for example:

    ```
    all:
    passwd group auto.home
    # hosts rpc services netid protocols mail \
    # netgrp shadow publickey networks ethers bootparams printcap \
    # amd.home auto.local. passwd.adjunct \
    # timezone locale netmasks
    ```

    This example allows NIS to create maps for the `/etc/passwd`, `/etc/group`, and `/etc/auto.home` files. By default, the information from the `/etc/shadow` file is merged with the `passwd` maps, and the information from the `/etc/gshadow` file is merged with the `group` maps.

    For more information, see the comments in `/var/yp/Makefile`.

    b. If you intend to use Kerberos authentication instead of NIS authentication, change the values of `MERGE_PASSWD` and `MERGE_GROUP` to `false`:

    ```
    MERGE_PASSWD=false
    MERGE_GROUP=false
    ```

    > **Note**
    >
    > These settings prevent password hashes from appearing in the NIS maps.

    c. If you configure any NIS slave servers in the domain, set the value of `NOPUSH` to `false`:

    ```
    NOPUSH=false
    ```

    If you update the maps, this setting allows the master server to automatically push the maps to the slave servers.

6. Configure the NIS services:

    a. Start the `ypserv` service and configure it to start after system reboots:

    ```
    # systemctl start ypserv
    # systemctl enable ypserv
    ```

    The `ypserv` service runs on the NIS master server and any slave servers.

    b. If the server will act as the master NIS server and there will be at least one slave NIS server, start the `ypxfrd` service and configure it to start after system reboots:

    ```
    # systemctl start ypxfrd
    # systemctl enable ypxfrd
    ```

    The `ypxfrd` service speeds up the distribution of very large NIS maps from an NIS master to any NIS slave servers. The service runs on the master server only, and not on any slave servers. You do not need to start this service if there are no slave servers.

    c. Start the `yppasswdd` service and configure it to start after system reboots:

    ```
    # systemctl start yppasswdd
    # systemctl enable yppasswdd
    ```

The `yppasswdd` service allows NIS users to change their password in the `shadow` map. The service runs on the NIS master server and any slave servers.

7. Configure the firewall settings:

a. Edit `/etc/sysconfig/network` and add the following entries that define the ports on which the `ypserv` and `ypxfrd` services listen:

```
YPSERV_ARGS="-p 834"
YPXFRD_ARGS="-p 835"
```

These entries fix the ports on which `ypserv` and `ypxfrd` listen.

b. Allow incoming TCP connections to ports 111 and 834 and incoming UDP datagrams on ports 111 and 834:

```
# firewall-cmd --zone=zone --add-port=111/tcp --add-port=111/udp \
  --add-port=834/tcp --add-port=834/udp
# firewall-cmd --permanent --zone=zone --add-port=111/tcp --add-port=111/udp \
  --add-port=834/tcp --add-port=834/udp
```

`portmapper` services requests on TCP port 111 and UDP port 111, and `ypserv` services requests on TCP port 834 and UDP port 834.

c. On the master server, if you run the `ypxfrd` service to support transfers to slave servers, allow incoming TCP connections to port 835 and incoming UDP datagrams on port 835:

```
# firewall-cmd --zone=zone --add-port=835/tcp --add-port=835/udp
# firewall-cmd --permanent --zone=zone --add-port=835/tcp --add-port=835/udp
```

d. Allow incoming UDP datagrams on the port on which `yppasswdd` listens:

```
# firewall-cmd --zone=zone \
  --add-port=`rpcinfo -p | gawk '/yppasswdd/ {print $4}'`/udp
```

> **Note**
>
> Do not make this rule permanent. The UDP port number that `yppasswdd` uses is different every time that it restarts.

e. Edit `/etc/rc.local` and add the following line:

```
firewall-cmd --zone=zone \
  --add-port=`rpcinfo -p | gawk '/yppasswdd/ {print $4}'`/udp
```

This entry creates a firewall rule for the `yppasswdd` service when the system reboots. If you restart `yppasswdd`, you must correct the firewall rules manually unless you modify the `/etc/init.d/yppasswdd` script.

8. After you have started all the servers, create the NIS maps on the master NIS server:

```
# /usr/lib64/yp/ypinit -m

At this point, we have to construct a list of the hosts which will run NIS
servers.  nismaster is in the list of NIS server hosts.  Please continue to add
the names for the other hosts, one per line.  When you are done with the
list, type a <control D>."
      next host to add:  nismaster
      next host to add:  nisslave1
      next host to add:  nisslave2
```

```
      next host to add:  ^D

The current list of NIS servers looks like this:

nismaster
nisslave1
nisslave2

Is this correct?  [y/n: y]  y
We need a few minutes to build the databases...
...
localhost has been set up as a NIS master server.

Now you can run ypinit -s nismaster on all slave server.
```

Enter the host names of the NIS slave servers (if any), type **Ctrl-D** to finish, and enter **y** to confirm the list of NIS servers. The host names must be resolvable to IP addresses in DNS or by entries in `/etc/hosts`.

The `ypinit` utility builds the domain subdirectory in `/var/yp` and makes the NIS maps that are defined for the `all` target in `/var/yp/Makefile`. If you have configured `NOPUSH=false` in `/var/yp/Makefile` and the names of the slave servers in `/var/yp/ypservers`, the command also pushes the updated maps to the slave servers.

9. On each NIS slave server, run the following command to initialize the server:

```
# /usr/lib64/yp/ypinit -s nismaster
```

where `nismaster` is the host name or IP address of the NIS master server.

For more information, see the `ypinit(8)` manual page

> **Note**
>
> If you update any of the source files on the master NIS server that are used to build the maps, use the following command on the master NIS server to remake the map and push the changes out to the slave servers:
>
> ```
> # make -C /var/yp
> ```

24.5.3 Adding User Accounts to NIS

> **Note**
>
> This procedure assumes that:
>
> - NIS provides maps for `passwd`, `group`, and `auto.home`.
>
> - The NIS master server uses NFS to export the users' home directories. See Section 22.2.2, "Mounting an NFS File System"

> **Warning**
>
> NIS authentication is deprecated as it has security issues, including a lack of protection of authentication data.

To create an account for an NIS user on the NIS master server:

1. If the NIS master server does not already export the base directory of the users' home directories, perform the following steps on the NIS master server:

a. Create the base directory for user directories, for example `/nethome`:

```
# mkdir /nethome
```

b. Add an entry such as the following to `/etc/exports`:

```
/nethome     *(rw,sync)
```

You might prefer to restrict which clients can mount the file system. For example, the following entry allows only clients in the 192.168.1.0/24 subnet to mount `/nethome`:

```
/nethome     192.168.1.0/24(rw,sync)
```

c. Use the following command to export the file system:

```
# exportfs -i -o ro,sync *:/nethome
```

d. If you have configured `/var/yp/Makfile` to make the `auto.home` map available to NIS clients, create the following entry in `/etc/auto.home`:

```
*    -rw,sync    nissvr:/nethome/&
```

where `nissvr` is the host name or IP address of the NIS server.

2. Create the user account:

```
# useradd -b /nethome username
```

The command updates the `/etc/passwd` file and creates a home directory on the NIS server.

3. Depending on the type of authentication that you have configured:

- For Kerberos authentication, on the Kerberos server or a client system with `kadmin` access, use `kadmin` to create a principal for the user in the Kerberos domain, for example:

```
# kadmin -q "addprinc username@KRBDOMAIN"
```

The command prompts you to set a password for the user, and adds the principal to the Kerberos database.

- For NIS authentication, use the `passwd` command:

```
# passwd username
```

The command updates the `/etc/shadow` file with the hashed password.

4. Update the NIS maps:

```
# make -C /var/yp
```

This command makes the NIS maps that are defined for the `all` target in `/var/yp/Makefile`. If you have configured `NOPUSH=false` in `/var/yp/Makefile` and the names of the slave servers in `/var/yp/ypservers`, the command also pushes the updated maps to the slave servers.

> **Note**
>
> A Kerberos-authenticated user can use either `kpasswd` or `passwd` to change his or her password. An NIS-authenticated user must use the `yppasswd` command rather than `passwd` to change his or her password.

24.5.4 Enabling NIS Authentication

To enable NIS authentication for an NIS client by using the Authentication Configuration GUI:

1. Install the `yp-tools` and `ypbind` packages:

   ```
   # yum install yp-tools ypbind
   ```

2. Run the Authentication Configuration GUI:

   ```
   # system-config-authentication
   ```

3. Select **NIS** as the user account database and enter values for:

NIS Domain	The name of the NIS domain. For example: `mynisdom`.
NIS Server	The domain name or IP address of the NIS server. For example, `nissvr.mydom.com`.

4. Select either **Kerberos password** or **NIS password** for authentication.

5. If you select Kerberos authentication, enter values for:

Realm	The name of the Kerberos realm.
KDCs	A comma-separated list of Key Distribution Center (KDC) servers that can issue Kerberos ticket granting tickets and service tickets.
Admin Servers	A comma-separated list of Kerberos administration servers.

   Alternatively, you can use DNS to configure these settings:

   - Select the **Use DNS to resolve hosts to realms** check box to look up the name of the realm defined as a `TXT` record in DNS, for example:

     ```
     _kerberos.mydom.com     IN TXT "MYDOM.COM"
     ```

   - Select the **Use DNS to locate KDCs for realms** check box to look up the KDCs and administration servers defined as `SVR` records in DNS, for example:

     ```
     _kerberos._tcp.mydom.com      IN SVR 1  0 88   krbsvr.mydom.com
     _kerberos._udp.mydom.com      IN SVR 1  0 88   krbsvr.mydom.com
     _kpasswd._udp.mydom.com       IN SVR 1  0 464  krbsvr.mydom.com
     _kerberos-adm._tcp.mydom.com  IN SVR 1  0 749  krbsvr.mydom.com
     ```

6. Click **Apply** to save your changes.

   Warning

   NIS authentication is deprecated as it has security issues, including a lack of protection of authentication data.

Figure 24.4 shows the Authentication Configuration GUI with NIS selected as the user account database and Kerberos selected for authentication.

Figure 24.4 Authentication Configuration of NIS with Kerberos Authentication

You can also enable and configure NIS or Kerberos authentication by using the `authconfig` command.

For example, to use NIS authentication, specify the `--enablenis` option together with the NIS domain name and the host name or IP address of the master server, as shown in the following example:.

```
# authconfig --enablenis --nisdomain mynisdom \
  --nisserver nissvr.mydom.com --update
```

The `--enablenis` option configures `/etc/nsswitch.conf` to enable the system to use NIS for information services. The `--nisdomain` and `--nisserver` settings are added to `/etc/yp.conf`.

For more information, see the `authconfig(8)`, `nsswitch.conf(5)`, and `yp.conf(5)` manual pages.

For information about using Kerberos authentication with NIS, see Section 24.6.3, "Enabling Kerberos Authentication".

24.5.4.1 Configuring an NIS Client to Use Automount Maps

If you have configured an automount map for `auto.home` in NIS, you can configure an NIS client to mount the users' home directories when they log in.

To configure an NIS client to automount users' home directories:

1. Install the `autofs` package:

```
# yum install autofs
```

2. Create an `/etc/auto.master` file that contains the following entry:

```
/nethome        /etc/auto.home
```

3. Verify that the `auto.home` map is available:

```
# ypcat -k auto.home
*    -rw,sync    nfssvr:/nethome/&
```

In this example, the map is available. For details of how to make this map available, see Section 24.5.3, "Adding User Accounts to NIS".

4. If the `auto.home` map is available, edit the file `/etc/auto.home` to contain the following entry:

```
+auto.home
```

This entry causes the automounter to use the `auto.home` map.

5. Restart the `autofs` service, and configure the service to start following a system reboot:

```
# systemctl restart autofs
# systemctl enable autofs
```

The `autofs` service creates the directory `/nethome`. When a user logs in, the automounter mounts his or her home directory under `/nethome`.

If the owner and group for the user's files are unexpectedly listed as the anonymous user or group (`nobody` or `nogroup`) and `all_squash` has not been specified as a mount option, verify that the `Domain` setting in `/etc/idmapd.conf` on the NFS server is set to the DNS domain name. Restart the NFS services on the NFS server if you change this file.

24.6 About Kerberos Authentication

Both LDAP and NIS authentication optionally support Kerberos authentication. In the case of IPA, Kerberos is fully integrated. Kerberos provides a secure connection over standard ports, and it also allows offline logins if you enable credential caching in SSSD.

Figure 24.5 illustrates how a Kerberos Key Distribution Center (KDC) authenticates a principal, which can be a user or a host, and grants a Ticket Granting Ticket (TGT) that the principal can use to gain access to a service.

Figure 24.5 Kerberos Authentication
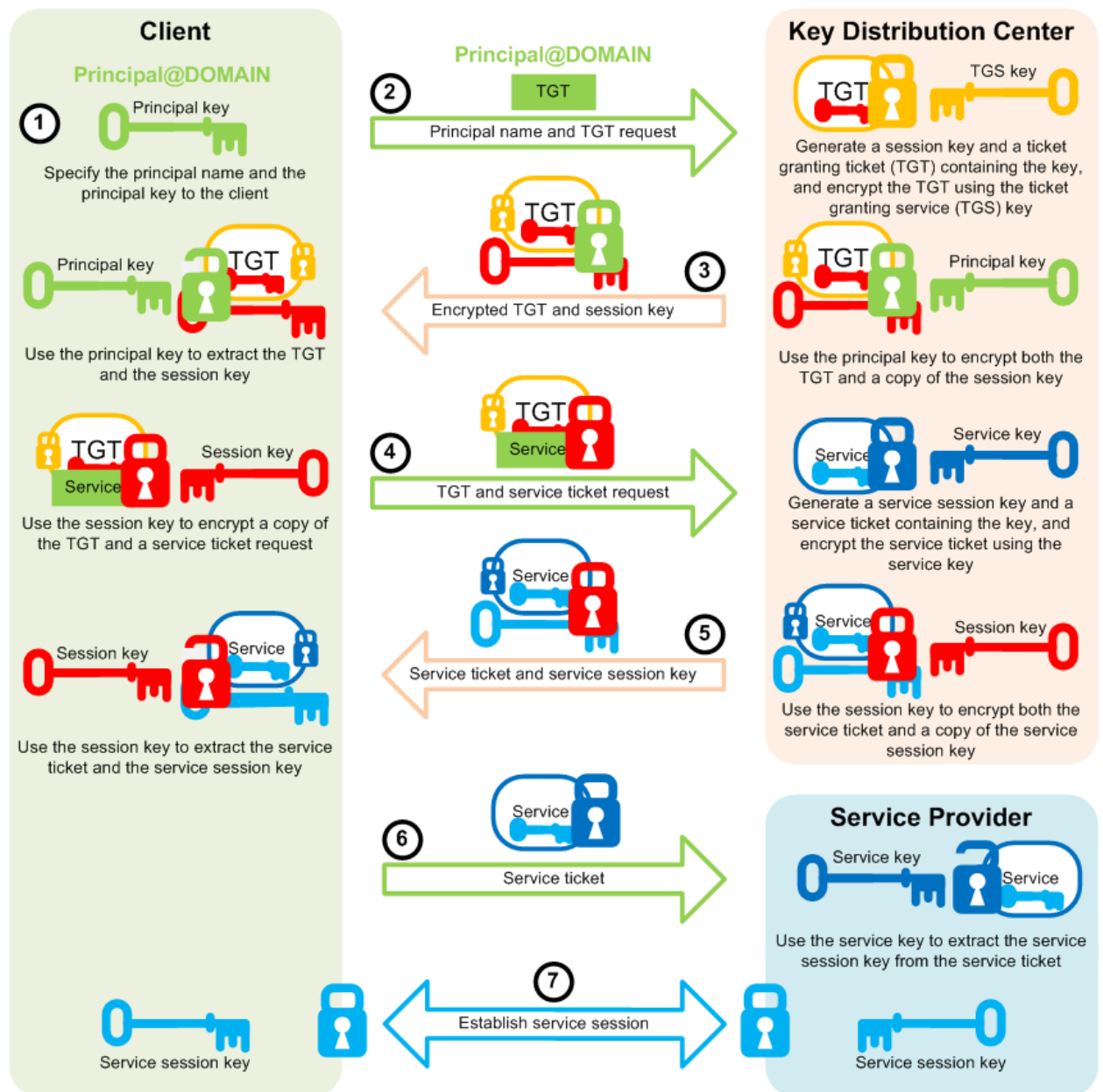
The steps in the process are:

1. A principal name and key are specified to the client.

2. The client sends the principal name and a request for a TGT to the KDC.

    The KDC generates a session key and a TGT that contains a copy of the session key, and uses the Ticket Granting Service (TGS) key to encrypt the TGT. It then uses the principal's key to encrypt both the already encrypted TGT and another copy of the session key.

3. The KDC sends the encrypted combination of the session key and the encrypted TGT to the client.

    The client uses the principal's key to extract the session key and the encrypted TGT.

4. When the client want to use a service, usually to obtain access to a local or remote host system, it uses the session key to encrypt a copy of the encrypted TGT, the client's IP address, a time stamp, and a service ticket request, and it sends this item to the KDC.

    The KDC uses its copies of the session key and the TGS key to extract the TGT, IP address, and time stamp, which allow it to validate the client. Provided that both the client and its service request are valid, the KDC generates a service session key and a service ticket that contains the client's IP address, a time stamp, and a copy of the service session key, and it uses the service key to encrypt the service ticket. It then uses the session key to encrypt both the service ticket and another copy of the service session key.

    The service key is usually the host principal's key for the system on which the service provider runs.

5. The KDC sends the encrypted combination of the service session key and the encrypted service ticket to the client.

    The client uses its copy of the session key to extract the encrypted service ticket and the service session key.

6. The client sends the encrypted service ticket to the service provider together with the principal name and a time stamp encrypted with the service session key.

    The service provider uses the service key to extract the data in the service session ticket, including the service session key.

7. The service provider enables the service for the client, which is usually to grant access to its host system.

    If the client and service provider are hosted on different systems, they can each use their own copy of the service session key to secure network communication for the service session.

Note the following points about the authentication handshake:

- Steps 1 through 3 correspond to using the `kinit` command to obtain and cache a TGT.

- Steps 4 through 7 correspond to using a TGT to gain access to a Kerberos-aware service.

- Authentication relies on pre-shared keys.

- Keys are never sent in the clear over any communications channel between the client, the KDC, and the service provider.

- At the start of the authentication process, the client and the KDC share the principal's key, and the KDC and the service provider share the service key. Neither the principal nor the service provider know the TGS key.

- At the end of the process, both the client and the service provider share a service session key that they can use to secure the service session. The client does not know the service key and the service provider does not know the principal's key.

- The client can use the TGT to request access to other service providers for the lifetime of the ticket, which is usually one day. The session manager renews the TGT if it expires while the session is active.

24.6.1 Configuring a Kerberos Server

If you want to configure any client systems to use Kerberos authentication, it is recommended that you first configure a Kerberos server. You can then configure any clients that you require.

Note

Keep any system that you configure as a Kerberos server very secure, and do not configure it to perform any other service function.

To configure a Kerberos server that can act as a key distribution center (KDC) and a Kerberos administration server:

1. Configure the server to use DNS and that both direct and reverse name lookups of the server's domain name and IP address work.

   For more information about configuring DNS, see Chapter 13, *Name Service Configuration*.

2. Configure the server to use network time synchronization mechanism such as the Network Time Protocol (NTP), Precision Time Protocol (PTP), or chrony. Kerberos requires that the system time on Kerberos servers and clients are synchronized as closely as possible. If the system times of the server and a client differ by more than 300 seconds (by default), authentication fails.

   For more information, see Chapter 14, *Network Time Configuration*.

3. Install the `krb5-libs`, `krb5-server`, and `krb5-workstation` packages:

   ```
   # yum install krb5-libs krb5-server krb5-workstation
   ```

4. Edit `/etc/krb5.conf` and configure settings for the Kerberos realm, for example:

   ```
   [logging]
    default = FILE:/var/log/krb5libs.log
    kdc = FILE:/var/log/krb5kdc.log
    admin_server = FILE:/var/log/kadmind.log

   [libdefaults]
    default_realm = MYDOM.COM
    dns_lookup_realm = false
    dns_lookup_kdc = false
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = true

   [realms]
    MYDOM.COM = {
     kdc = krbsvr.mydom.com
     admin_server = krbsvr.mydom.com
    }

   [domain_realm]
    .mydom.com = MYDOM.COM
    mydom.com = MYDOM.COM

   [appdefaults]
    pam = {
      debug = true
      validate = false
    }
   ```

   In this example, the Kerberos realm is `MYDOM.COM` in the DNS domain mydom.com and `krbsvr.mydom.com` (the local system) acts as both a KDC and an administration server. The `[appdefaults]` section configures options for the `pam_krb5.so` module.

   For more information, see the `krb5.conf(5)` and `pam_krb5(5)` manual pages.

5. Edit `/var/kerberos/krb5kdc/kdc.conf` and configure settings for the key distribution center, for example:

```
kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88

[realms]
 MYDOM.COM = {
  #master_key_type = aes256-cts
  master_key_type = des-hmac-sha1
  default_principal_flags = +preauth
  acl_file = /var/kerberos/krb5kdc/kadm5.acl
  dict_file = /usr/share/dict/words
  admin_keytab = /etc/kadm5.keytab
  supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal \
  arcfour-hmac:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal
 }
```

For more information, see the `kdc.conf(5)` manual page.

6. Create the Kerberos database and store the database password in a stash file:

```
# /usr/sbin/kdb5_util create -s
```

7. Edit `/var/kerberos/krb5kdc/kadm5.acl` and define the principals who have administrative access to the Kerberos database, for example:

```
*/admin@EXAMPLE.COM        *
```

In this example, any principal who has an instance of `admin`, such as `alice/admin@MYDOM.COM`, has full administrative control of the Kerberos database for the `MYDOM.COM` domain. Ordinary users in the database usually have an empty instance, for example `bob@MYDOM.COM`. These users have no administrative control other than being able to change their password, which is stored in the database.

8. Create a principal for each user who should have the `admin` instance, for example:

```
# kadmin.local -q "addprinc alice/admin"
```

9. Cache the keys that `kadmind` uses to decrypt administration Kerberos tickets in `/etc/kadm5.keytab`:

```
# kadmin.local -q "ktadd -k /etc/kadm5.keytab kadmin/admin"
# kadmin.local -q "ktadd -k /etc/kadm5.keytab kadmin/changepw"
```

10. Start the KDC and administration services and configure them to start following system reboots:

```
# systemctl start krb5kdc
# systemctl start kadmin
# systemctl enable krb5kdc
# systemctl enable kadmin
```

11. Add principals for users and the Kerberos server and cache the key for the server's host principal in `/etc/kadm5.keytab` by using either `kadmin.local` or `kadmin`, for example:

```
# kadmin.local -q "addprinc bob"
# kadmin.local -q "addprinc -randkey host/krbsvr.mydom.com"
# kadmin.local -q "ktadd -k /etc/kadm5.keytab host/krbsvr.mydom.com"
```

12. Allow incoming TCP connections to ports 88, 464, and 749 and UDP datagrams on UDP port 88, 464, and 749:

```
# firewall-cmd --zone=zone --add-port=88/tcp --add-port=88/udp \
```

```
    --add-port=464/tcp --add-port=464/udp \
    --add-port=749/tcp --add-port=749/udp
# firewall-cmd --permanent --zone=zone --add-port=88/tcp --add-port=88/udp \
    --add-port=464/tcp --add-port=464/udp \
    --add-port=749/tcp --add-port=749/udp
```

`krb5kdc` services requests on TCP port 88 and UDP port 88, and `kadmind` services requests on TCP ports 464 and 749 and UDP ports 464 and 749.

In addition, you might need to allow TCP and UDP access on different ports for other applications.

For more information, see the `kadmin(1)` manual page.

24.6.2 Configuring a Kerberos Client

Setting up a Kerberos client on a system allows it to use Kerberos to authenticate users who are defined in NIS or LDAP, and to provide secure remote access by using commands such as `ssh` with GSS-API enabled or the Kerberos implementation of `telnet`.

To set up a system as a Kerberos client:

1. Configure the client system to use DNS and that both direct and reverse name lookups of the domain name and IP address for both the client and the Kerberos server work.

   For more information about configuring DNS, see Chapter 13, *Name Service Configuration*.

2. Configure the system to use a network time synchronization protocol such as the Network Time Protocol (NTP). Kerberos requires that the system time on Kerberos servers and clients are synchronized as closely as possible. If the system times of the server and a client differ by more than 300 seconds (by default), authentication fails.

   To configure the server as an NTP client:

   a. Install the `ntp` package:

   ```
   # yum install ntp
   ```

   b. Edit `/etc/ntp.conf` and configure the settings as required. See the `ntp.conf(5)` manual page and http://www.ntp.org.

   c. Start the `ntpd` service and configure it to start following system reboots.

   ```
   # systemctl start ntpd
   # systemctl enable ntpd
   ```

3. Install the `krb5-libs` and `krb5-workstation` packages:

```
# yum install krb5-libs krb5-workstation
```

4. Copy the `/etc/krb5.conf` file to the system from the Kerberos server.

5. Use the Authentication Configuration GUI or `authconfig` to set up the system to use Kerberos with either NIS or LDAP, for example:

```
# authconfig --enablenis --enablekrb5 --krb5realm=MYDOM.COM \
    --krb5adminserver=krbsvr.mydom.com --krb5kdc=krbsvr.mydom.com \
    --update
```

See Section 24.6.3, "Enabling Kerberos Authentication".

---

6. On the Kerberos KDC, use either `kadmin` or `kadmin.local` to add a host principal for the client, for example:

```
# kadmin.local -q "addprinc -randkey host/client.mydom.com"
```

7. On the client system, use `kadmin` to cache the key for its host principal in `/etc/kadm5.keytab`, for example:

```
# kadmin -q "ktadd -k /etc/kadm5.keytab host/client.mydom.com"
```

8. To use `ssh` and related OpenSSH commands to connect from Kerberos client system to another Kerberos client system:

   a. On the remote Kerberos client system, verify that `GSSAPIAuthentication` is enabled in `/etc/ssh/sshd_config`:

   ```
   GSSAPIAuthentication yes
   ```

   b. On the local Kerberos client system, enable `GSSAPIAuthentication` and `GSSAPIDelegateCredentials` in the user's `.ssh/config` file:

   ```
   GSSAPIAuthentication yes
   GSSAPIDelegateCredentials yes
   ```

   Alternatively, the user can specify the `-K` option to `ssh`.

   c. Test that the principal can obtain a ticket and connect to the remote system, for example:

   ```
   $ kinit principal_name@MYDOM.COM
   $ ssh username@remote.mydom.com
   ```

   To allow use of the Kerberos versions of `rlogin`, `rsh`, and `telnet`, which are provided in the `krb5-appl-clients` package, you must enable the corresponding services on the remote client.

For more information, see the `kadmin(1)` manual page.

24.6.3 Enabling Kerberos Authentication

To be able to use Kerberos authentication with an LDAP or NIS client, use `yum` to install the `krb5-libs` and `krb5-workstation` packages.

If you use the Authentication Configuration GUI (`system-config-authentication`) and select LDAP or NIS as the user account database, select Kerberos password as the authentication method and enter values for:

Realm	The name of the Kerberos realm.
KDCs	A comma-separated list of Key Distribution Center (KDC) servers that can issue Kerberos ticket granting tickets and service tickets.
Admin Servers	A comma-separated list of Kerberos administration servers.

Alternatively, you can use DNS to configure these settings:

- Select the **Use DNS to resolve hosts to realms** check box to look up the name of the realm defined as a `TXT` record in DNS, for example:

```
_kerberos.mydom.com     IN TXT "MYDOM.COM"
```

- Select the **Use DNS to locate KDCs for realms** check box to look up the KDCs and administration servers defined as `SVR` records in DNS, for example:

```
_kerberos._tcp.mydom.com       IN SVR 1  0 88  krbsvr.mydom.com
_kerberos._udp.mydom.com       IN SVR 1  0 88  krbsvr.mydom.com
_kpasswd._udp.mydom.com        IN SVR 1  0 464 krbsvr.mydom.com
_kerberos-adm._tcp.mydom.com   IN SVR 1  0 749 krbsvr.mydom.com
```

Figure 24.6 shows the Authentication Configuration GUI with LDAP selected as the user account database and Kerberos selected for authentication.

Figure 24.6 Authentication Configuration of LDAP with Kerberos Authentication

Alternatively, you can use the `authconfig` command to configure Kerberos authentication with LDAP, for example:

```
# authconfig --enableldap \
  --ldapbasedn="dc=mydom,dc=com" --ldapserver=ldap://ldap.mydom.com:389 \
  [--enableldaptls --ldaploadcacert=https://ca-server.mydom.com/CAcert.pem] \
  --enablekrb5  \
  --krb5realm=MYDOM.COM | --enablekrb5realmdns \
  --krb5kdc=krbsvr.mydom.com --krb5adminserver=krbsvr.mydom.com | --enablekrb5kdcdns \
  --update
```

or with NIS:

```
# authconfig --enablenis \
  --enablekrb5 \
  --krb5realm=MYDOM.COM | --enablekrb5realmdns \
  --krb5kdc=krbsvr.mydom.com --krb5adminserver=krbsvr.mydom.com | --enablekrb5kdcdns \
  --update
```

The `--enablekrb5` option enables Kerberos authentication by modifying the PAM configuration files in `/etc/pam.d` to use the `pam_krb5.so` module. The `--enableldap` and `--enablenis` options configure `/etc/nsswitch.conf` to enable the system to use LDAP or NIS for information services.

For more information, see the `authconfig(8)`, `nsswitch.conf(5)`, and `pam_krb5(5)` manual pages.

24.7 About Pluggable Authentication Modules

The Pluggable Authentication Modules (PAM) feature is an authentication mechanism that allows you to configure how applications use authentication to verify the identity of a user. The PAM configuration files, which are located in the `/etc/pam.d` directory, describe the authentication procedure for an application. The name of each configuration file is the same as, or is similar to, the name of the application for which the module provides authentication. For example, the configuration files for `passwd` and `sudo` are named `passwd` and `sudo`.

Each PAM configuration file contains a list (*stack*) of calls to authentication modules. For example, the following listing shows the default content of the `login` configuration file:

```
#%PAM-1.0
auth [user_unknown=ignore success=ok ignore=ignore default=bad] pam_securetty.so
auth       include      system-auth
auth       include      postlogin
account    required     pam_nologin.so
account    include      system-auth
password   include      system-auth
# pam_selinux.so close should be the first session rule
session    required     pam_selinux.so close
session    required     pam_loginuid.so
session    optional     pam_console.so
# pam_selinux.so open should only be followed by sessions to be executed in the user context
session    required     pam_selinux.so open
session    required     pam_namespace.so
session    optional     pam_keyinit.so force revoke
session    include      system-auth
session    include      postlogin
-session   optional     pam_ck_connector.so
```

Comments in the file start with a `#` character. The remaining lines each define an operation type, a control flag, the name of a module such as `pam_rootok.so` or the name of an included configuration file such as `system-auth`, and any arguments to the module. PAM provides authentication modules as shared libraries in `/usr/lib64/security`.

For a particular operation type, PAM reads the stack from top to bottom and calls the modules listed in the configuration file. Each module generates a success or failure result when called.

The following operation types are defined for use:

`auth`	The module tests whether a user is authenticated or authorized to use a service or application. For example, the module might request and verify a password. Such modules can also set credentials, such as a group membership or a Kerberos ticket.
`account`	The module tests whether an authenticated user is allowed access to a service or application. For example, the module might check if a user account has expired or if a user is allowed to use a service at a given time.
`password`	The module handles updates to an authentication token.
`session`	The module configures and manages user sessions, performing tasks such as mounting or unmounting a user's home directory.

If the operation type is preceded with a dash (`-`), PAM does not add an create a system log entry if the module is missing.

With the exception of `include`, the control flags tell PAM what to do with the result of running a module. The following control flags are defined for use:

`optional`	The module is required for authentication if it is the only module listed for a service.
`required`	The module must succeed for access to be granted. PAM continues to execute the remaining modules in the stack whether the module succeeds or fails. PAM does not immediately inform the user of the failure.
`requisite`	The module must succeed for access to be granted. If the module succeeds, PAM continues to execute the remaining modules in the stack. However, if the module fails, PAM notifies the user immediately and does not continue to execute the remaining modules in the stack.
`sufficient`	If the module succeeds, PAM does not process any remaining modules of the same operation type. If the module fails, PAM processes the remaining modules of the same operation type to determine overall success or failure.

The control flag field can also define one or more rules that specify the action that PAM should take depending on the value that a module returns. Each rule takes the form *value=action*, and the rules are enclosed in square brackets, for example:

```
[user_unknown=ignore success=ok ignore=ignore default=bad]
```

If the result returned by a module matches a value, PAM uses the corresponding action, or, if there is no match, it uses the default action.

The `include` flag specifies that PAM must also consult the PAM configuration file specified as the argument.

Most authentication modules and PAM configuration files have their own manual pages. In addition, the `/usr/share/doc/pam-version` directory contains the PAM System Administrator's Guide (`html/Linux-PAM_SAG.html` or `Linux-PAM_SAG.txt`) and a copy of the PAM standard (`rfc86.0.txt`).

For more information, see the `pam(8)` manual page. In addition, each PAM module has its own manual page, for example `pam_unix(8)`, `postlogin(5)`, and `system-auth(5)`.

24.8 About the System Security Services Daemon

The System Security Services Daemon (SSSD) feature provides access on a client system to remote identity and authentication providers. The SSSD acts as an intermediary between local clients and any back-end provider that you configure.

The benefits of configuring SSSD include:

- Reduced system load

  Clients do not have to contact the identification or authentication servers directly.

- Offline authentication

  You can configure SSSD to maintain a cache of user identities and credentials.

- Single sign-on access

  If you configure SSSD to store network credentials, users need only authenticate once per session with the local system to access network resources.

For more information, see the `authconfig(8)`, `pam_sss(8)`, `sssd(8)`, and `sssd.conf(5)` manual pages and https://fedorahosted.org/sssd/.

24.8.1 Configuring an SSSD Server

To configure an SSSD server:

1. Install the `sssd` and `sssd-client` packages:

   ```
   # yum install sssd sssd-client
   ```

2. Edit the `/etc/sssd/sssd.conf` configuration file and configure the sections to support the required services, for example:

   ```
   [sssd]
   config_file_version = 2
   domains = LDAP
   services = nss, pam

   [domain/LDAP]
   id_provider = ldap
   ldap_uri = ldap://ldap.mydom.com
   ldap_search_base = dc=mydom,dc=com

   auth_provider = krb5
   krb5_server = krbsvr.mydom.com
   krb5_realm = MYDOM.COM
   cache_credentials = true

   min_id = 5000
   max_id = 25000
   enumerate = false

   [nss]
   filter_groups = root
   filter_users = root
   reconnection_retries = 3
   entry_cache_timeout = 300
   ```

```
[pam]
reconnection_retries = 3
offline_credentials_expiration = 2
offline_failed_login_attempts = 3
offline_failed_login_delay = 5
```

The `[sssd]` section contains configuration settings for SSSD monitor options, domains, and services. The SSSD monitor service manages the services that SSSD provides.

The `services` entry defines the supported services, which should include `nss` for the Name Service Switch and `pam` for Pluggable Authentication Modules.

The `domains` entry specifies the name of the sections that define authentication domains.

The `[domain/LDAP]` section defines a domain for an LDAP identity provider that uses Kerberos authentication. Each domain defines where user information is stored, the authentication method, and any configuration options. SSSD can work with LDAP identity providers such as OpenLDAP, Red Hat Directory Server, IPA, and Microsoft Active Directory, and it can use either native LDAP or Kerberos authentication.

The `id_provider` entry specifies the type of provider (in this example, LDAP). `ldap_uri` specifies a comma-separated list of the Universal Resource Identifiers (URIs) of the LDAP servers, in order of preference, to which SSSD can connect. `ldap_search_base` specifies the base distinguished name (`dn`) that SSSD should use when performing LDAP user operations on a relative distinguished name (RDN) such as a common name (`cn`).

The `auth_provider` entry specifies the authentication provider (in this example, Kerberos). `krb5_server` specifies a comma-separated list of Kerberos servers, in order of preference, to which SSSD can connect. `krb5_realm` specifies the Kerberos realm. `cache_credentials` specifies if SSSD caches user credentials such as tickets, session keys, and other identifying information to support offline authentication and single sign-on.

> **Note**
>
> To allow SSSD to use Kerberos authentication with an LDAP server, you must configure the LDAP server to use both Simple Authentication and Security Layer (SASL) and the Generic Security Services API (GSSAPI). For more information about configuring SASL and GSSAPI for OpenLDAP, see http://www.openldap.org/doc/admin24/sasl.html.

The `min_id` and `max_id` entries specify upper and lower limits on the values of user and group IDs. `enumerate` specifies whether SSSD caches the complete list of users and groups that are available on the provider. The recommended setting is `False` unless a domain contains relatively few users or groups.

The `[nss]` section configures the Name Service Switch (NSS) module that integrates the SSS database with NSS. The `filter_users` and `filter_groups` entries prevent NSS retrieving information about the specified users and groups being retrieved from SSS. `reconnection_retries` specifies the number of times that SSSD should attempt to reconnect if a data provider crashes. `enum_cache_timeout` specifies the number of seconds for which SSSD caches user information requests.

The `[pam]` section configures the PAM module that integrates SSS with PAM. The `offline_credentials_expiration` entry specifies the number of days for which to allow cached logins if the authentication provider is offline. `offline_failed_login_attempts`

specifies how many failed login attempts are allowed if the authentication provider is offline. `offline_failed_login_delay` specifies how many minutes after `offline_failed_login_attempts` failed login attempts that a new login attempt is permitted.

3. Change the mode of `/etc/sssd/sssd.conf` to 0600:

```
# chmod 0600 /etc/sssd/sssd.conf
```

4. Enable the SSSD service:

```
# authconfig --update --enablesssd --enablesssdauth
```

> **Note**
>
> If you edit `/etc/sssd/sssd.conf`, use this command to update the service.

The `--enablesssd` option updates `/etc/nsswitch.conf` to support SSS.

The `--enablesssdauth` option updates `/etc/pam.d/system-auth` to include the required `pam_sss.so` entries to support SSSD.

24.9 About Winbind Authentication

Winbind is a client-side service that resolves user and group information on a Windows server, and allows Oracle Linux to understand Windows users and groups. To be able to configure Winbind authentication, use `yum` to install the `samba-winbind` package. This package includes the `winbindd` daemon that implements the `winbind` service.

24.9.1 Enabling Winbind Authentication

If you use the Authentication Configuration GUI and select Winbind as the user account database, you are prompted for the information that is required to connect to a Microsoft workgroup, Active Directory, or Windows NT domain controller. Enter the name of the Winbind domain and select the security model for the Samba server:

`ads`
In the Activity Directory Server (ADS) security model, Samba acts as a domain member in an ADS realm, and clients use Kerberos tickets for Active Directory authentication. You must configure Kerberos and join the server to the domain, which creates a machine account for your server on the domain controller.

`domain`
In the domain security model, the local Samba server has a machine account (a domain security trust account) and Samba authenticates user names and passwords with a domain controller in a domain that implements Windows NT4 security.

> **Warning**
>
> If the local machine acts as a Primary or Backup Domain Controller, do not use the domain security model. Use the user security model instead.

`server`
In the server security model, the local Samba server authenticates user names and passwords with another server, such as a Windows NT server.

> ⊗ **Warning**
>
> The server security model is deprecated as it
> has numerous security issues.

`user` In the user security model, a client must log in with a valid user name
and password. This model supports encrypted passwords. If the server
successfully validates the client's user name and password, the client
can mount multiple shares without being required to specify a password.

Depending on the security model that you choose, you might also need to specify the following information:

* The name of the ADS realm that the Samba server is to join (ADS security model only).

* The names of the domain controllers. If there are several domain controllers, separate the names with
  spaces.

* The login template shell to use for the Windows NT user account (ADS and domain security models
  only).

* Whether to allow user authentication using information that has been cached by the System Security
  Services Daemon (SSSD) if the domain controllers are offline.

Your selection updates the security directive in the `[global]` section of the `/etc/samba/smb.conf`
configuration file.

If you have initialized Kerberos, you can click **Join Domain** to create a machine account on the Active
Directory server and grant permission for the Samba domain member server to join the domain.

You can also use the `authconfig` command to configure Winbind authentication. To use the user-
level security models, specify the name of the domain or workgroup and the host names of the domain
controllers. for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity user \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update
```

To allow user authentication using information that has been cached by the System Security Services
Daemon (SSSD) if the domain controllers are offline, specify the `--enablewinbindoffline` option.

For the domain security model, additionally specify the template shell, for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity domain \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update --winbindtemplateshell=/bin/bash --update
```

For the ADS security model, additionally specify the ADS realm and template shell, for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity ads \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update --smbrealm MYDOMAIN.COM \
  --winbindtemplateshell=/bin/bash --update
```

For more information, see the `authconfig(8)` manual page.

Chapter 25 Local Account Configuration

Table of Contents

This chapter describes how to configure and manage local user and group accounts.

25.1 About User and Group Configuration

You can use the User Manager GUI (`system-config-users`) to add or delete users and groups and to modify settings such as passwords, home directories, login shells, and group membership. Alternatively, you can use commands such as `useradd` and `groupadd`.

Figure 25.1 shows the User Manager GUI with the Users tab selected.

Figure 25.1 User Manager
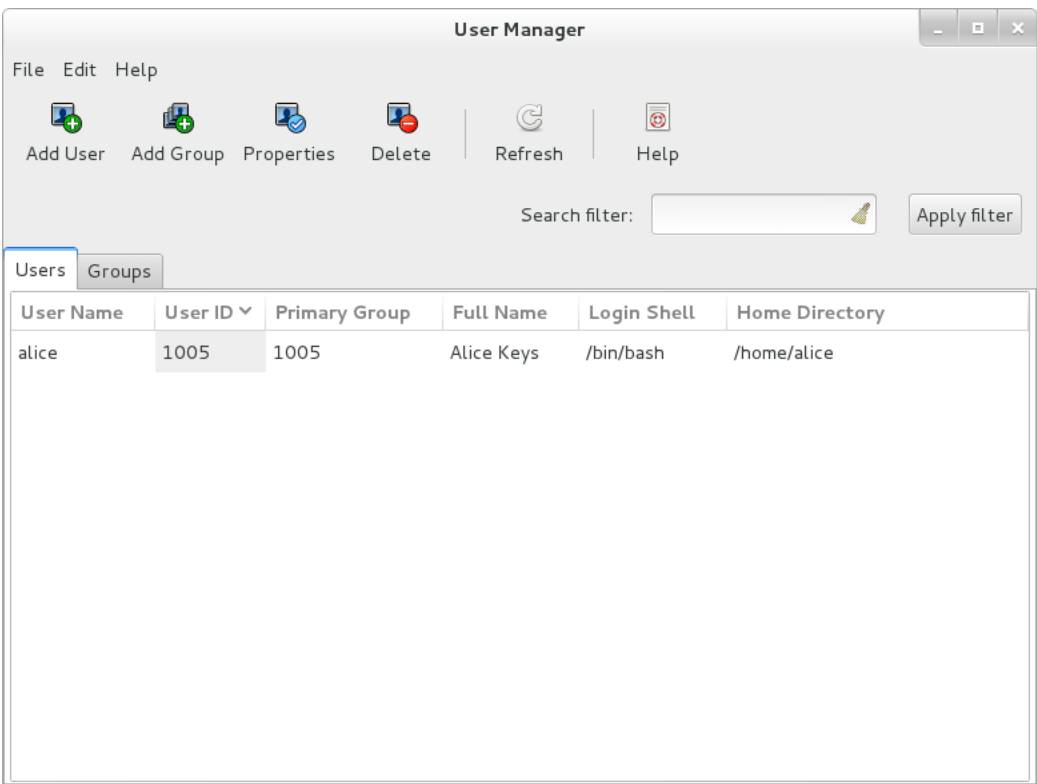
In an enterprise environment that might have hundreds of servers and thousands of users, user and group account information is more likely to be held in a central repository rather than in files on individual servers. You can configure user and group information on a central server and retrieve this information by using services such as Lightweight Directory Access Protocol (LDAP) or Network Information Service (NIS). You can also create users' home directories on a central server and automatically mount, or access, these remote file systems when a user logs in to a system.

25.2 Changing Default Settings for User Accounts

To display the default settings for an account use the following command:

```
# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes
```

INACTIVE specifies after how many days the system locks an account if a user's password expires. If set to 0, the system locks the account immediately. If set to -1, the system does not lock the account.

SKEL defines a template directory, whose contents are copied to a newly created user's home directory. The contents of this directory should match the default shell defined by SHELL.

You can specify options to useradd -D to change the default settings for user accounts. For example, to change the defaults for INACTIVE, HOME and SHELL:

```
# useradd -D -f 3 -b /home2 -s /bin/sh
```

> **Note**
>
> If you change the default login shell, you would usually also create a new SKEL template directory with contents that are appropriate to the new shell.
>
> If you specify /sbin/nologin for a user's SHELL, that user cannot log into the system directly but processes can run with that user's ID. This setting is typically used for services that run as users other than root.

The default settings are stored in the /etc/default/useradd file.

For more information, see Section 25.8, "Configuring Password Ageing" and the useradd(8) manual page.

25.3 Creating User Accounts

To create a user account by using the useradd command:

1. Enter the following command to create a user account:

   ```
   # useradd [options] username
   ```

   You can specify options to change the account's settings from the default ones.

   By default, if you specify a user name argument but do not specify any options, useradd creates a locked user account using the next available UID and assigns a user private group (UPG) rather than the value defined for GROUP as the user's group.

2. Assign a password to the account to unlock it:

```
# passwd username
```

The command prompts you to enter a password for the account.

If you want to change the password non-interactively (for example, from a script), use the `chpasswd` command instead:

```
echo "username:password" | chpasswd
```

Alternatively, you can use the `newusers` command to create a number of user accounts at the same time.

For more information, see the `chpasswd(8)`, `newusers(8)`, `passwd(1)`, and `useradd(8)` manual pages.

25.3.1 About umask and the setgid and Restricted Deletion Bits

Users whose primary group is not a UPG have a `umask` of 0022 set by `/etc/profile` or `/etc/bashrc`, which prevents other users, including other members of the primary group, from modifying any file that the user owns.

A user whose primary group is a UPG has a `umask` of 0002. It is assumed that no other user has the same group.

To grant users in the same group write access to files within the same directory, change the group ownership on the directory to the group, and set the `setgid` bit on the directory:

```
# chgrp groupname directory
# chmod g+s directory
```

Files created in such a directory have their group set to that of the directory rather than the primary group of the user who creates the file.

The restricted deletion bit prevents unprivileged users from removing or renaming a file in the directory unless they own either the file or the directory.

To set the restricted deletion bit on a directory:

```
# chmod a+t directory
```

For more information, see the `chmod(1)` manual page.

25.4 Locking an Account

To lock a user's account, enter:

```
# passwd -l username
```

To unlock the account:

```
# passwd -u username
```

For more information, see the `passwd(1)` manual page.

25.5 Modifying or Deleting User Accounts

To modify a user account, use the `usermod` command:

```
# usermod [options] username
```

For example, to add a user to a supplementary group (other than his or her login group):

```
# usermod -aG groupname username
```

You can use the `groups` command to display the groups to which a user belongs, for example:

```
# groups root
root : root bin daemon sys adm disk wheel
```

To delete a user's account, use the `userdel` command:

```
# userdel username
```

For more information, see the `groups(1)`, `userdel(8)` and `usermod(8)` manual pages.

25.6 Creating Groups

To create a group by using the `groupadd` command:

```
# groupadd [options] groupname
```

Typically, you might want to use the `-g` option to specify the group ID (GID). For example:

```
# groupadd -g 1000 devgrp
```

For more information, see the `groupadd(8)` manual page.

25.7 Modifying or Deleting Groups

To modify a group, use the `groupmod` command:

```
# groupmod [options] username
```

To delete a user's account, use the `groupdel` command:

```
# groupdel username
```

For more information, see the `groupdel(8)` and `groupmod(8)` manual pages.

25.8 Configuring Password Ageing

To specify how users' passwords are aged, edit the following settings in the `/etc/login.defs` file:

Setting	Description
PASS_MAX_DAYS	Maximum number of days for which a password can be used before it must be changed. The default value is 99,999 days.
PASS_MIN_DAYS	Minimum number of days that is allowed between password changes. The default value is 0 days.
PASS_WARN_AGE	Number of days warning that is given before a password expires. The default value is 7 days.

For more information, see the `login.defs(5)` manual page.

To change how long a user's account can be inactive before it is locked, use the `usermod` command. For example, to set the inactivity period to 30 days:

```
# usermod -f 30 username
```

To change the default inactivity period for new user accounts, use the `useradd` command:

```
# useradd -D -f 30
```

A value of -1 specifies that user accounts are not locked due to inactivity.

For more information, see the `useradd(8)` and `usermod(8)` manual pages.

25.9 Granting sudo Access to Users

By default, an Oracle Linux system is configured so that you cannot log in directly as `root`. You must log in as a named user before using either `su` or `sudo` to perform tasks as `root`. This configuration allows system accounting to trace the original login name of any user who performs a privileged administrative action. If you want to grant certain users authority to be able to perform specific administrative tasks via `sudo`, use the `visudo` command to modify the `/etc/sudoers` file.

For example, the following entry grants the user `erin` the same privileges as `root` when using `sudo`, but defines a limited set of privileges to `frank` so that he can run commands such as `systemctl`, `rpm`, and `yum`:

```
erin            ALL=(ALL)        ALL
frank           ALL= SERVICES, SOFTWARE
```

For more information, see the `su(1)`, `sudo(8)`, `sudoers(5)`, and `visudo(8)` manual pages.

Chapter 26 System Security Administration

Table of Contents

This chapter describes the subsystems that you can use to administer system security, including SELinux, the Netfilter firewall, TCP Wrappers, chroot jails, auditing, system logging, and process accounting.

26.1 About System Security

Oracle Linux provides a complete security stack, from network firewall control to access control security policies, and is designed to be secure by default.

Traditional Linux security is based on a Discretionary Access Control (DAC) policy, which provides minimal protection from broken software or from malware that is running as a normal user or as `root`. The SELinux enhancement to the Linux kernel implements the Mandatory Access Control (MAC) policy, which allows you to define a security policy that provides granular permissions for all users, programs, processes, files, and devices. The kernel's access control decisions are based on all the security relevant information available, and not solely on the authenticated user identity. By default, SELinux is enabled when you install an Oracle Linux system.

Oracle Linux has evolved into a secure enterprise-class operating system that can provide the performance, data integrity, and application uptime necessary for business-critical production environments.

Thousands of production systems at Oracle run Oracle Linux and numerous internal developers use it as their development platform. Oracle Linux is also at the heart of several Oracle engineered systems, including the Oracle Exadata Database Machine, Oracle Exalytics In-Memory Machine, Oracle Exalogic Elastic Cloud, and Oracle Database Appliance.

Oracle On Demand services, which deliver software as a service (SaaS) at a customer's site, via an Oracle data center, or at a partner site, use Oracle Linux at the foundation of their solution architectures. Backed by Oracle support, these mission-critical systems and deployments depend fundamentally on the built-in security and reliability features of the Oracle Linux operating system.

Released under an open-source license, Oracle Linux includes the Unbreakable Enterprise Kernel that provides the latest Linux innovations while offering tested performance and stability. Oracle has been a key participant in the Linux community, contributing code enhancements such as Oracle Cluster File System and the Btrfs file system. From a security perspective, having roots in open source is a significant advantage. The Linux community, which includes many experienced developers and security experts, reviews posted Linux code extensively prior to its testing and release. The open-source Linux community has supplied many security improvements over time, including access control lists (ACLs), cryptographic libraries, and trusted utilities.

26.2 Configuring and Using SELinux

Traditional Linux security is based on a Discretionary Access Control (DAC) policy, which provides minimal protection from broken software or from malware that is running as a normal user or as `root`. Access to files and devices is based solely on user identity and ownership. Malware or broken software can do anything with files and resources that the user that started the process can do. If the user is `root` or the application is `setuid` or `setgid` to `root`, the process can have `root`-access control over the entire file system.

The National Security Agency created Security Enhanced Linux (SELinux) to provide a finer-grained level of control over files, processes, users and applications in the Linux operating system. The SELinux enhancement to the Linux kernel implements the Mandatory Access Control (MAC) policy, which allows you to define a security policy that provides granular permissions for all users, programs, processes, files, and devices. The kernel's access control decisions are based on all the security relevant information available, and not solely on the authenticated user identity.

When security-relevant access occurs, such as when a process attempts to open a file, SELinux intercepts the operation in the kernel. If a MAC policy rule allows the operation, it continues; otherwise, SELinux blocks the operation and returns an error to the process. The kernel checks and enforces DAC policy rules before MAC rules, so it does not check SELinux policy rules if DAC rules have already denied access to a resource.

The following table describes the SELinux packages that are installed by default with Oracle Linux:

Package	Description
policycoreutils	Provides utilities such as `load_policy`, `restorecon`, `secon`, `setfiles`, `semodule`, `sestatus`, and `setsebool` for operating and managing SELinux.
libselinux	Provides the API that SELinux applications use to get and set process and file security contexts, and to obtain security policy decisions.
selinux-policy	Provides the SELinux Reference Policy, which is used as the basis for other policies, such as the SELinux targeted policy.

Package	Description
`selinux-policy-targeted`	Provides support for the SELinux targeted policy, where objects outside the targeted domains run under DAC.
`libselinux-python`	Contains Python bindings for developing SELinux applications.
`libselinux-utils`	Provides the `avcstat`, `getenforce`, `getsebool`, `matchpathcon`, `selinuxconlist`, `selinuxdefcon`, `selinuxenabled`, `setenforce`, and `togglesebool` utilities.

The following table describes a selection of useful SELinux packages that are not installed by default:

Package	Description
`mcstrans`	Translates SELinux levels, such as `s0-s0:c0.c1023`, to an easier-to-read form, such as `SystemLow-SystemHigh`.
`policycoreutils-gui`	Provides a GUI (`system-config-selinux`) that you can use to manage SELinux. For example, you can use the GUI to set the system default enforcing mode and policy type.
`policycoreutils-python`	Provides additional Python utilities for operating SELinux, such as `audit2allow`, `audit2why`, `chcat`, and `semanage`.
`selinux-policy-mls`	Provides support for the strict Multilevel Security (MLS) policy as an alternative to the SELinux targeted policy.
`setroubleshoot`	Provides the GUI that allows you to view `setroubleshoot-server` messages using the `sealert` command.
`setroubleshoot-server`	Translates access-denial messages from SELinux into detailed descriptions that you can view on the command line using the `sealert` command.
`setools-console`	Provides the Tresys Technology SETools distribution of tools and libraries, which you can use to analyze and query policies, monitor and report audit logs, and manage file context.

Use `yum` or another suitable package manager to install the SELinux packages that you require on your system.

For more information about SELinux, refer to the SELinux Project Wiki, the `selinux(8)` manual page, and the manual pages for the SELinux commands.

26.2.1 About SELinux Administration

The following table describes the utilities that you can use to administer SELinux, and the packages that contain each utility.

Utility	Package	Description
`audit2allow`	`policycoreutils-python`	Generates SELinux policy `allow_audit` rules from logs of denied operations.
`audit2why`	`policycoreutils-python`	Generates SELinux policy `don't_audit` rules from logs of denied operations.
`avcstat`	`libselinux-utils`	Displays statistics for the SELinux Access Vector Cache (AVC).
`chcat`	`policycoreutils-python`	Changes or removes the security category for a file or user.
`findcon`	`setools-console`	Searches for file context.

Utility	Package	Description
fixfiles	policycoreutils	Fixes the security context for file systems.
getenforce	libselinux-utils	Reports the current SELinux mode.
getsebool	libselinux-utils	Reports SELinux boolean values.
indexcon	setools-console	Indexes file context.
load_policy	policycoreutils	Loads a new SELinux policy into the kernel.
matchpathcon	libselinux-utils	Queries the system policy and displays the default security context that is associated with the file path.
replcon	setools-console	Replaces file context.
restorecon	policycoreutils	Resets the security context on one or more files.
restorecond	policycoreutils	Daemon that watches for file creation and sets the default file context.
sandbox	policycoreutils-python	Runs a command in an SELinux sandbox.
sealert	setroubleshoot-server, setroubleshoot	Acts as the user interface to the setroubleshoot system, which diagnoses and explains SELinux AVC denials and provides recommendations on how to prevent such denials.
seaudit-report	setools-console	Reports from the SELinux audit log.
sechecker	setools-console	Checks SELinux policies.
secon	policycoreutils	Displays the SELinux context from a file, program, or user input.
sediff	setools-console	Compares SELinux polices.
seinfo	setools-console	Queries SELinux policies.
selinuxconlist	libselinux-utils	Displays all SELinux contexts that are reachable by a user.
selinuxdefcon	libselinux-utils	Displays the default SELinux context for a user.
selinuxenabled	libselinux-utils	Indicates whether SELinux is enabled.
semanage	policycoreutils-python	Manages SELinux policies.
semodule	policycoreutils	Manages SELinux policy modules.
semodule_deps	policycoreutils	Displays the dependencies between SELinux policy packages.
semodule_expand	policycoreutils	Expands a SELinux policy module package.
semodule_link	policycoreutils	Links SELinux policy module packages together.
semodule_package	policycoreutils	Creates a SELinux policy module package.
sesearch	setools-console	Queries SELinux policies.
sestatus	policycoreutils	Displays the SELinux mode and the SELinux policy that are in use.
setenforce	libselinux-utils	Modifies the SELinux mode.
setsebool	policycoreutils	Sets SELinux boolean values.
setfiles	policycoreutils	Sets the security context for one or more files.

Utility	Package	Description
`system-config-selinux`	`policycoreutils-gui`	Provides a GUI that you can use to manage SELinux.
`togglesebool`	`libselinux-utils`	Flips the current value of an SELinux boolean.

26.2.2 About SELinux Modes

SELinux runs in one of three modes.

`Disabled`   The kernel uses only DAC rules for access control. SELinux does not enforce any security policy because no policy is loaded into the kernel.

`Enforcing`   The kernel denies access to users and programs unless permitted by SELinux security policy rules. All denial messages are logged as AVC (Access Vector Cache) denials. This is the default mode that enforces SELinux security policy.

`Permissive`   The kernel does not enforce security policy rules but SELinux sends denial messages to a log file. This allows you to see what actions would have been denied if SELinux were running in enforcing mode. This mode is intended to used for diagnosing the behavior of SELinux.

26.2.3 Setting SELinux Modes

You can set the default and current SELinux mode in the Status view of the SELinux Administration GUI (`system-config-selinux`).

Alternatively, to display the current mode, use the `getenforce` command:

```
# getenforce
Enforcing
```

To set the current mode to `Enforcing`, enter:

```
# setenforce Enforcing
```

To set the current mode to `Permissive`, enter:

```
# setenforce Permissive
```

The current value that you set for a mode using `setenforce` does not persist across reboots. To configure the default SELinux mode, edit the configuration file for SELinux, `/etc/selinux/config`, and set the value of the `SELINUX` directive to `disabled`, `enabled`, or `permissive`.

26.2.4 About SELinux Policies

An SELinux policy describes the access permissions for all users, programs, processes, and files, and for the devices upon which they act. You can configure SELinux to implement either Targeted Policy or Multilevel Security (MLS) Policy.

26.2.4.1 Targeted Policy

Applies access controls to a limited number of processes that are believed to be most likely to be the targets of an attack on the system. Targeted processes run in their own SELinux domain, known as a

confined domain, which restricts access to files that an attacker could exploit. If SELinux detects that a targeted process is trying to access resources outside the confined domain, it denies access to those resources and logs the denial. Only specific services run in confined domains. Examples are services that listen on a network for client requests, such as `httpd`, `named`, and `sshd`, and processes that run as `root` to perform tasks on behalf of users, such as `passwd`. Other processes, including most user processes, run in an unconfined domain where only DAC rules apply. If an attack compromises an unconfined process, SELinux does not prevent access to system resources and data.

The following table lists examples of SELinux domains.

Domain	Description
`init_t`	`systemd`
`httpd_t`	HTTP daemon threads
`kernel_t`	Kernel threads
`syslogd_t`	`journald` and `rsyslogd` logging daemons
`unconfined_t`	Processes executed by Oracle Linux users run in the unconfined domain

26.2.4.2 Multilevel Security (MLS) Policy

Applies access controls to multiple levels of processes with each level having different rules for user access. Users cannot obtain access to information if they do not have the correct authorization to run a process at a specific level. In SELinux, MLS implements the Bell-LaPadula (BLP) model for system security, which applies labels to files, processes and other system objects to control the flow of information between security levels. In a typical implementation, the labels for security levels might range from the most secure, `top secret`, through `secret`, and `classified`, to the least secure, `unclassified`. For example, under MLS, you might configure a program labelled `secret` to be able to write to a file that is labelled `top secret`, but not to be able to read from it. Similarly, you would permit the same program to read from and write to a file labelled `secret`, but only to read `classified` or `unclassified` files. As a result, information that passes through the program can flow upwards through the hierarchy of security levels, but not downwards.

> **Note**
>
> You must install the `selinux-policy-mls` package if you want to be able to apply the MLS policy.

26.2.4.3 Setting SELinux Policies

> **Note**
>
> You cannot change the policy type of a running system.

You can set the default policy type in the **Status** view of the SELinux Administration GUI.

Alternatively, to configure the default policy type, edit `/etc/selinux/config` and set the value of the `SELINUXTYPE` directive to `targeted` or `mls`.

26.2.4.4 Customizing SELinux Policies

You can customize an SELinux policy by enabling or disabling the members of a set of boolean values. Any changes that you make take effect immediately and do not require a reboot.

You can set the boolean values in the **Boolean** view of the SELinux Administration GUI.

Alternatively, to display all boolean values together with a short description, use the following command:

```
# semanage boolean -l
SELinux boolean              State  Default Description

ftp_home_dir                  (off  ,  off)
Determine whether ftpd can read and write files in user home directories.
smartmon_3ware                (off  ,  off)
Determine whether smartmon can support devices on 3ware controllers.
mpd_enable_homedirs           (off  ,  off)
Determine whether mpd can traverse user home directories.
...
```

You can use the `getsebool` and `setsebool` commands to display and set the value of a specific boolean.

```
# getsebool boolean
# setsebool boolean on|off
```

For example, to display and set the value of the `ftp_home_dir` boolean:

```
# getsebool ftp_home_dir
ftp_home_dir --> off
# setsebool ftp_home_dir on
# getsebool ftp_home_dir
ftp_home_dir --> on
```

To toggle the value of a boolean, use the `togglesebool` command as shown in this example:

```
# togglesebool ftp_home_dir
ftp_home_dir: inactive
```

To make the value of a boolean persist across reboots, specify the `-P` option to `setsebool`, for example:

```
# setsebool -P ftp_home_dir on
# getsebool ftp_home_dir
ftp_home_dir --> on
```

26.2.5 About SELinux Context

Under SELinux, all file systems, files, directories, devices, and processes have an associated security context. For files, SELinux stores a context label in the extended attributes of the file system. The context contains additional information about a system object: the SELinux user, their role, their type, and the security level. SELinux uses this context information to control access by processes, Linux users, and files.

You can specify the `-Z` option to certain commands (`ls`, `ps`, and `id`) to display the SELinux context with the following syntax:

```
SELinux user:Role:Type:Level
```

where the fields are as follows:

SELinux user	An SELinux user account compliments a regular Linux user account. SELinux maps every Linux user to an SELinux user identity that is used in the SELinux context for the processes in a user session.
Role	In the Role-Based Access Control (RBAC) security model, a role acts as an intermediary abstraction layer between SELinux process domains

or file types and an SELinux user. Processes run in specific SELinux domains, and file system objects are assigned SELinux file types. SELinux users are authorized to perform specified roles, and roles are authorized for specified SELinux domains and file types. A user's role determines which process domains and file types he or she can access, and hence, which processes and files, he or she can access.

`Type` A type defines an SELinux file type or an SELinux process domain. Processes are separated from each other by running in their own domains. This separation prevents processes from accessing files that other processes use, and prevents processes from accessing other processes. The SELinux policy rules define the access that process domains have to file types and to other process domains.

`Level` A level is an attribute of Multilevel Security (MLS) and Multicategory Security (MCS). An MLS range is a pair of sensitivity levels, written as `low_level-high_level`. The range can be abbreviated as `low_level` if the levels are identical. For example, `s0` is the same as `s0-s0`. Each level has an optional set of security categories to which it applies. If the set is contiguous, it can be abbreviated. For example, `s0:c0.c3` is the same as `s0:c0,c1,c2,c3`.

26.2.5.1 Displaying SELinux User Mapping

To display the mapping between SELinux and Linux user accounts, select the User Mapping view in the the SELinux Administration GUI.

Alternatively, enter the following command to display the user mapping:

```
# semanage login -l

Login Name          SELinux User         MLS/MCS Range        Service

__default__         unconfined_u         s0-s0:c0.c1023       *
root                unconfined_u         s0-s0:c0.c1023       *
system_u            system_u             s0-s0:c0.c1023       *
```

By default, SELinux maps Linux users other than `root` and the default system-level user, `system_u`, to the Linux `__default__` user, and in turn to the SELinux `unconfined_u` user. The MLS/MCS Range is the security level used by Multilevel Security (MLS) and Multicategory Security (MCS).

26.2.5.2 Displaying SELinux Context Information

To display the context information that is associated with files, use the `ls -Z` command:

```
# ls -Z
-rw-------. root root system_u:object_r:admin_home_t:s0 anaconda-ks.cfg
-rw-r--r--. root root unconfined_u:object_r:admin_home_t:s0 config
-rw-r--r--. root root system_u:object_r:admin_home_t:s0 initial-setup-ks.cfg
drwxr-xr-x. root root unconfined_u:object_r:admin_home_t:s0 jail
-rw-r--r--. root root unconfined_u:object_r:admin_home_t:s0 team0.cfg
```

To display the context information that is associated with a specified file or directory:

```
# ls -Z /etc/selinux/config
-rw-r--r--. root root system_u:object_r:selinux_config_t:s0 /etc/selinux/config
```

To display the context information that is associated with processes, use the `ps -Z` command:

```
# ps -Z
LABEL                                            PID  TTY    TIME     CMD
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3038 pts/0 00:00:00 su
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3044 pts/0 00:00:00 bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3322 pts/0 00:00:00 ps
```

To display the context information that is associated with the current user, use the `id -Z` command:

```
# id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

26.2.5.3 Changing the Default File Type

Under some circumstances, you might need to change the default file type for a file system hierarchy. For example, you might want to use a `DocumentRoot` directory other than `/var/www/html` with `httpd`.

To change the default file type of the directory hierarchy `/var/webcontent` to `httpd_sys_content_t`:

1. Use the `semanage` command to define the file type `httpd_sys_content_t` for the directory hierarchy:

   ```
   # /usr/sbin/semanage fcontext -a -t httpd_sys_content_t "/var/webcontent(/.*)?"
   ```

   This command adds the following entry to the file `/etc/selinux/targeted/contexts/files/file_contexts.local`:

   ```
   /var/webcontent(/.*)?    system_u:object_r:httpd_sys_content_t:s0
   ```

2. Use the `restorecon` command to apply the new file type to the entire directory hierarchy.

   ```
   # /sbin/restorecon -R -v /var/webcontent
   ```

26.2.5.4 Restoring the Default File Type

To restore the default file type of the directory hierarchy `/var/webcontent` after previously changing it to `httpd_sys_content_t`:

1. Use the `semanage` command to delete the file type definition for the directory hierarchy from the file `/etc/selinux/targeted/contexts/files/file_contexts.local`:

   ```
   # /usr/sbin/semanage fcontext -d "/var/webcontent(/.*)?"
   ```

2. Use the `restorecon` command to apply the default file type to the entire directory hierarchy.

   ```
   # /sbin/restorecon -R -v /var/webcontent
   ```

26.2.5.5 Relabelling a File System

If you see an error message that contains the string `file_t`, the problem usually lies with a file system having an incorrect context label.

To relabel a file system, use one of the following methods:

- In the **Status** view of the SELinux Administration GUI, select the **Relabel on next reboot** option.

- Create the file `/.autorelabel` and reboot the system.

- Run the `fixfiles onboot` command and reboot the system.

26.2.6 About SELinux Users

As described in Section 26.2.5, "About SELinux Context", each SELinux user account compliments a regular Oracle Linux user account. SELinux maps every Oracle Linux user to an SELinux user identity that is used in the SELinux context for the processes in a user session.

SELinux users form part of a SELinux policy that is authorized for a specific set of roles and for a specific MLS (Multi-Level Security) range, and each Oracle Linux user is mapped to an SELinux user as part of the policy. As a result, Linux users inherit the restrictions and security rules and mechanisms placed on SELinux users. To define the roles and levels of users, the mapped SELinux user identity is used in the SELinux context for processes in a session. You can display user mapping in the **User Mapping** view of the SELinux Administration GUI. You can also view the mapping between SELinux and Oracle Linux user accounts from the command line:

```
# semanage login -l
Login Name     SELinux User     MLS/MCS Range
_default_      unconfined_u     s0-s0:c0.c1023
root           unconfined_u     s0-s0:c0.c1023
system_u       system_u         s0-s0:c0.c1023
```

The MLS/MCS Range column displays the level used by MLS and MCS.

By default, Oracle Linux users are mapped to the SELinux user `unconfined_u`.

You can configure SELinux to confine Oracle Linux users by mapping them to SELinux users in confined domains, which have predefined security rules and mechanisms as listed in the following table.

SELinux User	SELinux Domain	Permit Running su and sudo?	Permit Network Access?	Permit Logging in Using X Window System?	Permit Executing Applications in $HOME and /tmp?
guest_u	guest_t	No	Yes	No	No
staff_u	staff_t	sudo	Yes	Yes	Yes
system_u	ssystem_t	Yes	Yes	Yes	Yes
user_u	user_t	No	Yes	Yes	Yes
xguest_x	xguest_t	No	Firefox only	Yes	No

26.2.6.1 Mapping Oracle Linux Users to SELinux Users

To map an Oracle Linux user `oluser` to an SELinux user such as `user_u`, use the `semanage` command:

```
# semanage login -a -s user_u oluser
```

26.2.6.2 Configuring the Behavior of Application Execution for Users

To help prevent flawed or malicious applications from modifying a user's files, you can use booleans to specify whether users are permitted to run applications in directories to which they have write access, such as in their home directory hierarchy and /tmp.

To allow Oracle Linux users in the `guest_t` and `xguest_t` domains to execute applications in directories to which they have write access:

```
# setsebool -P allow_guest_exec_content on
```

```
# setsebool -P allow_xguest_exec_content on
```

To prevent Linux users in the `staff_t` and `user_t` domains from executing applications in directories to which they have write access:

```
# setsebool -P allow_staff_exec_content off
# setsebool -P allow_user_exec_content off
```

26.2.7 Troubleshooting Access-Denial Messages

The decisions that SELinux has made about allowing denying access are stored in the Access Vector Cache (AVC). If the auditing service (`auditd`) is not running, SELinux logs AVC denial messages to `/var/log/messages`. Otherwise, the messages are logged to `/var/log/audit/audit.log`. If the `setroubleshootd` daemon is running, easier-to-read versions of the denial messages are also written to `/var/log/messages`.

If you have installed the `setroubleshoot` and `setroubleshoot-server` packages, the `auditd` and `setroubleshoot` services are running, and you are using the X Window System, you can use the `sealert -b` command to run the SELinux Alert Browser, which displays information about SELinux AVC denials. To view the details of the alert, click **Show**. To view a recommended solution, click **Troubleshoot**.

If you do not use the SELinux Alert Browser, you can search in `/var/log/audit/audit.log` for messages containing the string `denied`, and in `/var/log/messages` for messages containing the string `SELinux is preventing`. For example:

```
# grep denied /var/log/audit/audit.log
type=AVC msg=audit(1364486257.632:26178): avc:  denied  { read } for
pid=5177 comm="httpd" name="index.html" dev=dm-0 ino=396075
scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:acct_data_t:s0 tclass=file
```

The main causes of access-denial problems are:

- The context labels for an application or file are incorrect.

  A solution might be to change the default file type of the directory hierarchy. For example, change the default file type from `/var/webcontent` to `httpd_sys_content_t`:

  ```
  # /usr/sbin/semanage fcontext -a -t httpd_sys_content_t "/var/webcontent(/.*)?"
  # /sbin/restorecon -R -v /var/webcontent
  ```

- A Boolean that configures a security policy for a service is set incorrectly.

  A solution might be to change the value of a Boolean. For example, allow users' home directories to be browsable by turning on `httpd_enable_homedirs`:

  ```
  # setsebool -P httpd_enable_homedirs on
  ```

- A service attempts to access a port to which a security policy does not allow access.

  If the service's use of the port is valid, a solution is to use `semanage` to add the port to the policy configuration. For example, allow the Apache HTTP server to listen on port 8000:

  ```
  # semanage port -a -t http_port_t -p tcp 8000
  ```

- An update to a package causes an application to behave in a way that breaks an existing security policy.

  You can use the `audit2allow -w -a` command to view the reason why an access denial occurred.

If you then run the `audit2allow -a -M module` command, it creates a type enforcement (`.te`) file and a policy package (`.pp`) file. You can use the policy package file with the `semodule -i module.pp` command to stop the error from reoccurring. This procedure is usually intended to allow package updates to function until an amended policy is available. If used incorrectly, it can create potential security holes on your system.

26.3 About Packet-filtering Firewalls

A packet filtering firewall filters incoming and outgoing network packets based on the packet header information. You can create packet filter rules that determine whether packets are accepted or rejected. For example, if you create a rule to block a port, any request is made to that port that is blocked by the firewall, and the request is ignored. Any service that is listening on a blocked port is effectively disabled.

The Oracle Linux kernel uses the Netfilter feature to provide packet filtering functionality for IPv4 and IPv6 packets.

Netfilter consists of two components:

- A `netfilter` kernel component consisting of a set of tables in memory for the rules that the kernel uses to control network packet filtering.

- Utilities to create, maintain, and display the rules that `netfilter` stores. In Oracle Linux 7, the default firewall utility is `firewall-cmd`, which is provided by the `firewalld` package.

  If you prefer, you can enable the `iptables` and `iptables6` services and use the `iptables` and `ip6tables` utilities, provided by the `iptables` package. These were the default utilities for firewall configuration in Oracle Linux 6.

The `firewalld`-based firewall has the following advantages over an `iptables`-based firewall:

- Unlike the `iptables` and `ip6tables` commands, using `firewalld-cmd` does not restart the firewall and disrupt established TCP connections.

- `firewalld` supports dynamic zones, which allow you to implement different sets of firewall rules for systems such as laptops that can connect to networks with different levels of trust. You are unlikely to use this feature with server systems.

- `firewalld` supports D-Bus for better integration with services that depend on firewall configuration.

To implement a general-purpose firewall, you can use the Firewall Configuration GUI (`firewall-config`), provided by the `firewall-config` package.

Figure 26.1 shows the Firewall Configuration GUI.

Figure 26.1 Firewall Configuration
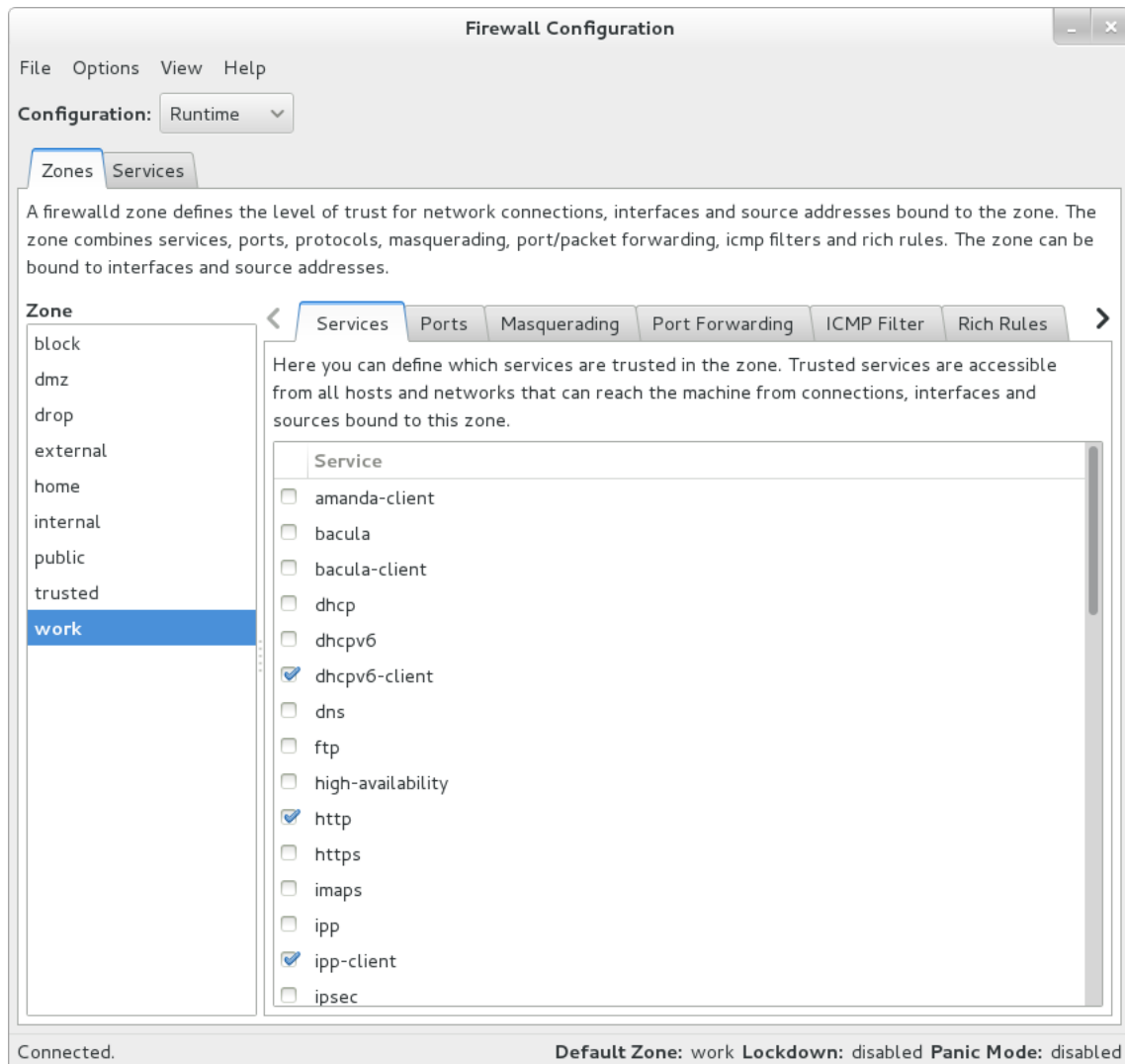
To create or modify a firewall configuration from the command line, use the `firewall-cmd` utility (or, if you prefer, the `iptables`, or `ip6tables` utilities) to configure the packet filtering rules.

The packet filtering rules are recorded in the `/etc/firewalld` hierarchy for `firewalld` and in the `/etc/sysconfig/iptables` and `/etc/sysconfig/ip6tables` files for `iptables` and `ip6tables`.

26.3.1 Controlling the firewalld Firewall Service

The `firewalld` service is enabled by default in Oracle Linux 7. You can use the `systemctl` command to start, stop, or restart the service, and to query its status.

26.3.1.1 Configuring the firewalld Zone

To check the zone for which your system's firewall is configured:

```
# firewall-cmd --get-active-zone
```

The command does not display any results if the system has not been assigned to a zone.

Use the following command to display all available zones:

```
# firewall-cmd --get-zones
block dmz drop external home internal public trusted work
```

To configure your system for the `work` zone on a local network connected via the `em1` interface:

```
# firewall-cmd --zone=work --change-interface=em1
success
```

Querying the current zone now shows that the firewall is configured on the interface `em1` for the `work` zone:

```
# firewall-cmd --get-active-zone
work
  interfaces: em1
```

To make the change permanent, you can change the default zone for the system, for example:

```
# firewall-cmd --get-default-zone
public
# firewall-cmd --set-default-zone=work
success
# firewall-cmd --get-default-zone
work
```

26.3.1.2 Controlling Access to Services

You can permit or deny access to a service by specifying its name. The following command lists the services to which access is allowed on the local system for the `work` zone:

```
# firewall-cmd --zone=work --list-services
ssh samba
```

In this example, the system allows access by SSH and Samba clients.

To permit access by NFS and HTTP clients when the `work` zone is active, use the `--add-service` option:

```
# firewall-cmd --zone=work --add-service=http --add-service=nfs
success
# firewall-cmd --zone=work --list-services
http nfs ssh samba
```

> **Note**
>
> If you do not specify the zone, the change is applied to the default zone, not the currently active zone.

To make rule changes persist across reboots, run the command again, additionally specifying the `--permanent` option:

```
# firewall-cmd --permanent --zone=work --add-service=http --add-service=nfs
success
```

To remove access to a service, use the `--remove-service` option, for example:

```
# firewall-cmd --zone=work --remove-service=samba
success
# firewall-cmd --permanent --zone=work --remove-service=samba
success
# firewall-cmd --zone=work --list-services
http nfs ssh
```

26.3.1.3 Controlling Access to Ports

You can permit or deny access to a port by specifying the port number and the associated protocol. The `--list-port` option lists the ports and associated protocols to which you have explicitly allowed access, for example:

```
# firewall-cmd --zone=work --list-ports
3689/tcp
```

You can use the `--add-port` option to permit access:

```
# firewall-cmd --zone=work --add-port=5353/udp
success
# firewall-cmd --permanent --zone=work --add-port=5353/udp
success
# firewall-cmd --zone=work --list-ports
5353/udp 3689/tcp
```

Similarly, the `--remove-port` option removes access to a port. Remember to rerun the command with the `--permanant` option if you want to make the change persist.

To display all the firewall rules that are defined for a zone, use the `--list-all` option:

```
# firewall-cmd --zone=work --list-all
work (default,active)
  interfaces: em1
  sources:
  services: http nfs ssh
  ports: 5353/udp 3689/tcp
  masquerade: no
  forward-ports:
  icmp-blocks:
  rich rules:
```

For more information, see the `firewall-cmd(1)` manual page.

26.3.2 Controlling the iptables Firewall Service

If you want to use `iptables` instead of `firewalld`, first stop and disable the `firewalld` service before starting the `iptables` firewall service and enabling it to start when the system boots:

```
# systemctl stop firewalld
# systemctl disable firewalld
# systemctl start iptables
# systemctl enable iptables
```

To save any changes that you have made to the firewall rules to `/etc/sysconfig/iptables`, so that the service loads them when it next starts:

```
# /sbin/iptables-save > /etc/sysconfig/iptables
```

To restart the service so that it re-reads its rules from `/etc/sysconfig/iptables`:

```
# systemctl restart iptables
```

To stop the service:

```
# systemctl stop iptables
```

To control IPv6 filtering, use `ip6tables` instead of `iptables`.

For more information, see the `iptables(8)`, and `ip6tables(8)` manual pages.

26.3.2.1 About netfilter Tables Used by iptables and ip6tables

The `netfilter` tables used by `iptables` and `ip6tables` include:

Filter   The default table, which is mainly used to drop or accept packets based on their content.

Mangle   This table is used to alter certain fields in a packet.

NAT   The Network Address Translation table is used to route packets that create new connections.

The kernel uses the rules stored in these tables to make decisions about network packet filtering. Each rule consists of one or more criteria and a single action. If a criterion in a rule matches the information in a network packet header, the kernel applies the action to the packet. Examples of actions include:

ACCEPT   Continue processing the packet.

DROP   End the packet's life without notice.

REJECT   As `DROP`, and additionally notify the sending system that the packet was blocked.

Rules are stored in chains, where each chain is composed of a default policy plus zero or more rules. The kernel applies each rule in a chain to a packet until a match is found. If there is no matching rule, the kernel applies the chain's default action (policy) to the packet.

Each `netfilter` table has several predefined chains. The `filter` table contains the following chains:

FORWARD   Packets that are not addressed to the local system pass through this chain.

INPUT   Inbound packets to the local system pass through this chain.

OUTPUT   Locally created packets pass through this chain.

The chains are permanent and you cannot delete them. However, you can create additional chains in the filter table.

26.3.2.2 Listing Firewall Rules

Use the `iptables -L` command to list firewall rules for the chains of the `filter` table. The following example shows the default rules for a newly installed system:

```
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source        destination
ACCEPT     all  --  anywhere      anywhere        state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere      anywhere
ACCEPT     all  --  anywhere      anywhere
ACCEPT     tcp  --  anywhere      anywhere        state NEW tcp dpt:ssh
ACCEPT     udp  --  anywhere      anywhere        state NEW udp dpt:ipp
ACCEPT     udp  --  anywhere      224.0.0.251     state NEW udp dpt:mdns
ACCEPT     tcp  --  anywhere      anywhere        state NEW tcp dpt:ipp
ACCEPT     udp  --  anywhere      anywhere        state NEW udp dpt:ipp
REJECT     all  --  anywhere      anywhere        reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source        destination
REJECT     all  --  anywhere      anywhere        reject-with icmp-host-prohibited
```

```
Chain OUTPUT (policy ACCEPT)
target     prot opt source        destination
```

In this example, the default policy for each chain is `ACCEPT`. A more secure system could have a default policy of `DROP`, and the additional rules would only allow specific packets on a case-by-case basis.

If you want to modify the chains, specify the `--line-numbers` option to see how the rules are numbered.

```
# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source        destination
1    ACCEPT     all  --  anywhere      anywhere      state RELATED,ESTABLISHED
2    ACCEPT     icmp --  anywhere      anywhere
3    ACCEPT     all  --  anywhere      anywhere
4    ACCEPT     tcp  --  anywhere      anywhere      state NEW tcp dpt:ssh
5    ACCEPT     udp  --  anywhere      anywhere      state NEW udp dpt:ipp
6    ACCEPT     udp  --  anywhere      224.0.0.251   state NEW udp dpt:mdns
7    ACCEPT     tcp  --  anywhere      anywhere      state NEW tcp dpt:ipp
8    ACCEPT     udp  --  anywhere      anywhere      state NEW udp dpt:ipp
9    REJECT     all  --  anywhere      anywhere      reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source        destination
1    REJECT     all  --  anywhere      anywhere      reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source        destination
```

26.3.2.3 Inserting and Replacing Rules in a Chain

Use the `iptables -I` command to insert a rule in a chain. For example, the following command inserts a rule in the `INPUT` chain to allow access by TCP on port 80:

```
# iptables -I INPUT 4 -p tcp -m tcp --dport 80 -j ACCEPT
# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source        destination
1    ACCEPT     all  --  anywhere      anywhere      state RELATED,ESTABLISHED
2    ACCEPT     icmp --  anywhere      anywhere
3    ACCEPT     all  --  anywhere      anywhere
4    ACCEPT     tcp  --  anywhere      anywhere      tcp dpt:http
5    ACCEPT     tcp  --  anywhere      anywhere      state NEW tcp dpt:ssh
6    ACCEPT     udp  --  anywhere      anywhere      state NEW udp dpt:ipp
7    ACCEPT     udp  --  anywhere      224.0.0.251   state NEW udp dpt:mdns
8    ACCEPT     tcp  --  anywhere      anywhere      state NEW tcp dpt:ipp
9    ACCEPT     udp  --  anywhere      anywhere      state NEW udp dpt:ipp
10   REJECT     all  --  anywhere      anywhere      reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source        destination
1    REJECT     all  --  anywhere      anywhere      reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source        destination
```

The output from `iptables -L` shows that the new entry has been inserted as rule 4, and the old rules 4 through 9 are pushed down to positions 5 through 10. The TCP destination port of 80 is represented as `http`, which corresponds to the following definition in the `/etc/services` file (the HTTP daemon listens for client requests on port 80):

```
http            80/tcp          www www-http    # WorldWideWeb HTTP
```

To replace the rule in a chain, use the `iptables -R` command. For example, the following command replaces rule 4 in the `INPUT` chain to allow access by TCP on port 443:

```
# iptables -I INPUT 4 -p tcp -m tcp --dport 443 -j ACCEPT
# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target      prot opt source              destination
1    ACCEPT      all  --  anywhere            anywhere            state RELATED,ESTABLISHED
2    ACCEPT      icmp --  anywhere            anywhere
3    ACCEPT      all  --  anywhere            anywhere
4    ACCEPT      tcp  --  anywhere            anywhere            tcp dpt:https
...
```

The TCP destination port of 443 is represented as `https`, which corresponds to the following definition in the `/etc/services` file for secure HTTP on port 443:

```
https           443/tcp                         # http protocol over TLS/SSL
```

26.3.2.4 Deleting Rules in a Chain

Use the `iptables -D` command to delete a rule in a chain. For example, the following command deletes rule 4 from the `INPUT` chain:

```
# iptables -D INPUT 4
```

To delete all rules in a chain, enter:

```
# iptables -F chain
```

To delete all rules in all chains, enter:

```
# iptables -F
```

26.3.2.5 Saving Rules

To save your changes to the firewall rules so that they are loaded when the `iptables` service next starts, use the following command:

```
# /sbin/iptables-save /etc/sysconfig/iptables
```

The command saves the rules to `/etc/sysconfig/iptables`. For IPv6, you can use `/sbin/ip6tables-save > /etc/sysconfig/ip6tables` to save the rules to `/etc/sysconfig/ip6tables`.

26.4 About TCP Wrappers

TCP wrappers provide basic filtering of incoming network traffic. You can allow or deny access from other systems to certain *wrapped* network services running on a Linux server. A wrapped network service is one that has been compiled against the `libwrap.a` library. You can use the `ldd` command to determine if a network service has been wrapped as shown in the following example for the `sshd` daemon:

```
# ldd /usr/sbin/sshd | grep libwrap
 libwrap.so.0 => /lib64/libwrap.so.0 (0x00007f877de07000)
```

When a remote client attempts to connect to a network service on the system, the wrapper consults the rules in the configuration files `/etc/hosts.allow` and `/etc/hosts.deny` files to determine if access is permitted.

The wrapper for a service first reads `/etc/hosts.allow` from top to bottom. If the daemon and client combination matches an entry in the file, access is allowed. If the wrapper does not find a match in `/etc/hosts.allow`, it reads `/etc/hosts.deny` from top to bottom. If the daemon and client combination matches and entry in the file, access is denied. If no rules for the daemon and client combination are found in either file, or if neither file exists, access to the service is allowed.

The wrapper first applies the rules specified in `/etc/hosts.allow`, so these rules take precedence over the rules specified in `/etc/hosts.deny`. If a rule defined in `/etc/hosts.allow` permits access to a service, any rule in `/etc/hosts.deny` that forbids access to the same service is ignored.

The rules take the following form:

```
daemon_list : client_list [: command] [: deny]
```

where `daemon_list` and `client_list` are comma-separated lists of daemons and clients, and the optional `command` is run when a client tries to access a daemon. You can use the keyword `ALL` to represent all daemons or all clients. Subnets can be represented by using the `*` wildcard, for example `192.168.2.*`. Domains can be represented by prefixing the domain name with a period (`.`), for example `.mydomain.com`. The optional `deny` keyword causes a connection to be denied even for rules specified in the `/etc/hosts.allow` file.

The following are some sample rules.

Match all clients for `scp`, `sftp`, and `ssh` access (`sshd`).

```
sshd : ALL
```

Match all clients on the 192.168.2 subnet for FTP access (`vsftpd`).

```
vsftpd : 192.168.2.*
```

Match all clients in the `mydomain.com` domain for access to all wrapped services.

```
ALL : .mydomain.com
```

Match all clients for FTP access, and displays the contents of the banner file `/etc/banners/vsftpd` (the banner file must have the same name as the daemon).

```
vsftpd : ALL : banners /etc/banners/
```

Match all clients on the 200.182.68 subnet for all wrapped services, and logs all such events. The `%c` and `%d` tokens are expanded to the names of the client and the daemon.

```
ALL : 200.182.68.* : spawn /usr/bin/echo `date` "Attempt by %c to connect to %d" >> /var/log/tcpwr.log
```

Match all clients for `scp`, `sftp`, and `ssh` access, and logs the event as an `emerg` message, which is displayed on the console.

```
sshd : ALL : severity emerg
```

Match all clients in the `forbid.com` domain for `scp`, `sftp`, and `ssh` access, logs the event, and deny access (even if the rule appears in `/etc/hosts.allow`).

```
sshd : .forbid.com : spawn /usr/bin/echo `date` "sshd access denied for %c" >>/var/log/sshd.log : deny
```

For more information, see the `hosts_access(5)` manual page.

26.5 About chroot Jails

A `chroot` operation changes the apparent root directory for a running process and its children. It allows you to run a program with a root directory other than `/`. The program cannot see or access files outside the designated directory tree. Such an artificial root directory is called a *chroot jail*, and its purpose is to limit the directory access of a potential attacker. The chroot jail locks down a given process and any user ID that it is using so that all they see is the directory in which the process is running. To the process, it appears that the directory in which it is running is the root directory.

> **Note**
>
> The `chroot` mechanism cannot defend against intentional tampering or low-level access to system devices by privileged users. For example, a `chroot root` user could create device nodes and mount file systems on them. A program can also break out of a chroot jail if it can gain `root` privilege and use `chroot()` to change its current working directory to the real `root` directory. For this reason, you should ensure that a chroot jail does not contain any `setuid` or `setgid` executables that are owned by `root`.

For a `chroot` process to be able to start successfully, you must populate the `chroot` directory with all required program files, configuration files, device nodes, and shared libraries at their expected locations relative to the level of the `chroot` directory.

26.5.1 Running DNS and FTP Services in a Chroot Jail

If the DNS name service daemon (`named`) runs in a chroot jail, any hacker that enters your system via a BIND exploit is isolated to the files under the chroot jail directory. Installing the `bind-chroot` package creates the `/var/named/chroot` directory, which becomes the chroot jail for all BIND files.

You can configure the `vsftpd` FTP server to automatically start chroot jails for clients. By default, anonymous users are placed in a chroot jail. However, local users that access an `vsftpd` FTP server are placed in their home directory. Specify the `chroot_local_user=YES` option in the `/etc/vsftpd/vsftpd.conf` file to place local users in a chroot jail based on their home directory.

26.5.2 Creating a Chroot Jail

To create a chroot jail:

1. Create the directory that will become the root directory of the chroot jail, for example:

   ```
   # mkdir /home/oracle/jail
   ```

2. Use the `ldd` command to find out which libraries are required by the command that you intend to run in the chroot jail, for example `/usr/bin/bash`:

   ```
   # ldd /usr/bin/bash
    linux-vdso.so.1 =>  (0x00007fffdedfe000)
    libtinfo.so.5 => /lib64/libtinfo.so.5 (0x0000003877000000)
    libdl.so.2 => /lib64/libdl.so.2 (0x0000003861c00000)
    libc.so.6 => /lib64/libc.so.6 (0x0000003861800000)
    /lib64/ld-linux-x86-64.so.2 (0x0000003861000000)
   ```

   > **Note**
   >
   > Although the path is displayed as `/lib64`, the actual path is `/usr/lib64` because `/lib64` is a symbolic link to `/usr/lib64`. Similarly, `/bin` is a symbolic link to `/usr/bin`. You need to recreate such symbolic links within the chroot jail.

3. Create subdirectories of the chroot jail's root directory that have the same relative paths as the command binary and its required libraries have to the real root directory, for example:

   ```
   # mkdir -p /home/oracle/jail/usr/bin
   # mkdir -p /home/oracle/jail/usr/lib64
   ```

4. Create the symbolic links that link to the binary and library directories in the same manner as the symbolic links that exists in the real root directory.

```
# ln -s /home/oracle/jail/usr/bin /home/oracle/jail/bin
# ln -s /home/oracle/jail/usr/lib64 /home/oracle/jail/lib64
```

5. Copy the binary and the shared libraries to the directories under the chroot jail's root directory, for example:

```
# cp /usr/bin/bash /home/oracle/jail/usr/bin
# cp /usr/lib64/{libtinfo.so.5,libdl.so.2,libc.so.6,ld-linux-x86-64.so.2} \
  /home/oracle/jail/usr/lib64
```

26.5.3 Using a Chroot Jail

To run a command in a chroot jail in an existing directory (`chroot_jail`), use the following command:

```
# chroot chroot_jail command
```

If you do not specify a command argument, `chroot` runs the value of the `SHELL` environment variable or `/usr/bin/sh` if `SHELL` is not set.

For example, to run `/usr/bin/bash` in a chroot jail (having previously set it up as described in Section 26.5.2, "Creating a Chroot Jail"):

```
# chroot /home/oracle/jail
bash-4.2# pwd
/
bash-4.2# ls
bash: ls: command not found
bash-4.2# exit
exit
#
```

You can run built-in shell commands such as `pwd` in this shell, but not other commands unless you have copied their binaries and any required shared libraries to the chroot jail.

For more information, see the `chroot(1)` manual page.

26.6 About Auditing

Auditing collects data at the kernel level that you can analyze to identify unauthorized activity. Auditing collects more data in greater detail than system logging, but most audited events are uninteresting and insignificant. The process of examining audit trails to locate events of interest can be a significant challenge that you will probably need to automate.

The audit configuration file, `/etc/audit/auditd.conf`, defines the data retention policy, the maximum size of the audit volume, the action to take if the capacity of the audit volume is exceeded, and the locations of local and remote audit trail volumes. The default audit trail volume is `/var/log/audit/audit.log`. For more information, see the `auditd.conf(5)` manual page.

By default, auditing captures specific events such as system logins, modifications to accounts, and `sudo` actions. You can also configure auditing to capture detailed system call activity or modifications to certain files. The kernel audit daemon (`auditd`) records the events that you configure, including the event type, a time stamp, the associated user ID, and success or failure of the system call.

The entries in the audit rules file, `/etc/audit/audit.rules`, determine which events are audited. Each rule is a command-line option that is passed to the `auditctl` command. You should typically configure this file to match your site's security policy.

The following are examples of rules that you might set in the `/etc/audit/audit.rules` file.

Record all unsuccessful exits from `open` and `truncate` system calls for files in the `/etc` directory hierarchy.

```
-a exit,always -S open -S truncate -F /etc -F success=0
```

Record all files opened by a user with UID 10.

```
-a exit,always -S open -F uid=10
```

Record all files that have been written to or that have their attributes changed by any user who originally logged in with a UID of 500 or greater.

```
-a exit,always -S open -F auid>=500 -F perm=wa
```

Record requests for write or file attribute change access to `/etc/sudoers`, and tag such record with the string `sudoers-change`.

```
-w /etc/sudoers -p wa -k sudoers-change
```

Record requests for write and file attribute change access to the `/etc` directory hierarchy.

```
-w /etc/ -p wa
```

Require a reboot after changing the audit configuration. If specified, this rule should appear at the end of the `/etc/audit/audit.rules` file.

```
-e 2
```

You can find more examples of audit rules in `/usr/share/doc/audit-version/stig.rules`, and in the `auditctl(8)` and `audit.rules(7)` manual pages.

Stringent auditing requirements can impose a significant performance overhead and generate large amounts of audit data. Some site security policies stipulate that a system must shut down if events cannot be recorded because the audit volumes have exceeded their capacity. As a general rule, you should direct audit data to separate file systems in rotation to prevent overspill and to facilitate backups.

You can use the `-k` option to tag audit records so that you can locate them more easily in an audit volume with the `ausearch` command. For example, to examine records tagged with the string `sudoers-change`, you would enter:

```
# ausearch -k sudoers-change
```

The `aureport` command generates summaries of audit data. You can set up `cron` jobs that run `aureport` periodically to generate reports of interest. For example, the following command generates a reports that shows every login event from 1 second after midnight on the previous day until the current time:

```
# aureport -l -i -ts yesterday -te now
```

For more information, see the `ausearch(8)` and `aureport(8)` manual pages.

26.7 About System Logging

The log files contain messages about the system, kernel, services, and applications. The `journald` logging daemon, which is part of `systemd`, records system messages in non-persistent journal files in memory and in the `/run/log/journal` directory. `journald` forwards messages to the system logging daemon, `rsyslog`. As files in `/run` are volatile, the log data is lost after a reboot unless you create the directory `/var/log/journal`. You can use the `journalctl` command to query the journal logs.

For more information, see the `journalctl(1)` and `systemd-journald.service(8)` manual pages.

The configuration file for `rsyslogd` is `/etc/rsyslog.conf`, which contains global directives, module directives, and rules. By default, `rsyslog` processes and archives only `syslog` messages. If required, you can configure `rsyslog` to archive any other messages that `journald` forwards, including kernel, boot, `initrd`, `stdout`, and `stderr` messages.

Global directives specify configuration options that apply to the `rsyslogd` daemon. All configuration directives must start with a dollar sign (`$`) and only one directive can be specified on each line. The following example specifies the maximum size of the `rsyslog` message queue:

```
$MainMsgQueueSize 50000
```

The available configuration directives are described in the file `/usr/share/doc/rsyslog-version-number/rsyslog_conf_global.html`.

The design of `rsyslog` allows its functionality to be dynamically loaded from modules, which provide configuration directives. To load a module, specify the following directive:

```
$ModLoad MODULE_name
```

Modules have the following main categories:

- Input modules gather messages from various sources. Input module names always start with the `im` prefix (examples include `imfile` and `imrelp`).

- Filter modules allow `rsyslogd` to filter messages according to specified rules. The name of a filter module always starts with the `fm` prefix.

- Library modules provide functionality for other loadable modules. `rsyslogd` loads library modules automatically when required. You cannot configure the loading of library modules.

- Output modules provide the facility to store messages in a database or on other servers in a network, or to encrypt them. Output module names always starts with the `om` prefix (examples include `omsnmp` and `omrelp`).

- Message modification modules change the content of an `rsyslog` message.

- Parser modules allow `rsyslogd` to parse the message content of messages that it receives. The name of a parser module always starts with the `pm` prefix.

- String generator modules generate strings based on the content of messages in cooperation with `rsyslog`'s template feature. The name of a string generator module always starts with the `sm` prefix.

Input modules receive messages, which pass them to one or more parser modules. A parser module creates a representation of a message in memory, possibly modifying the message, and passes the internal representation to output modules, which can also modify the content before outputting the message.

A description of the available modules can be found at http://www.rsyslog.com/doc/rsyslog_conf_modules.html.

An `rsyslog` rule consists of a filter part, which selects a subset of messages, and an action part, which specifies what to do with the selected messages. To define a rule in the `/etc/rsyslog.conf` configuration file, specify a filter and an action on a single line, separated by one or more tabs or spaces.

You can configure `rsyslog` to filter messages according to various properties. The most commonly used filters are:

- Expression-based filters, written in the `rsyslog` scripting language, select messages according to arithmetic, boolean, or string values.

- Facility/priority-based filters filter messages based on facility and priority values that take the form *facility.priority*.

- Property-based filters filter messages by properties such as `timegenerated` or `syslogtag`.

The following table lists the available facility keywords for facility/priority-based filters:

Facility Keyword	Description
`auth`, `authpriv`	Security, authentication, or authorization messages.
`cron`	`crond` messages.
`daemon`	Messages from system daemons other than `crond` and `rsyslogd`.
`kern`	Kernel messages.
`lpr`	Line printer subsystem.
`mail`	Mail system.
`news`	Network news subsystem.
`syslog`	Messages generated internally by `rsyslogd`.
`user`	User-level messages.
`UUCP`	UUCP subsystem.
`local0` - `local7`	Local use.

The following table lists the available priority keywords for facility/priority-based filters, in ascending order of importance:

Priority Keyword	Description
`debug`	Debug-level messages.
`info`	Informational messages.
`notice`	Normal but significant condition.
`warning`	Warning conditions.
`err`	Error conditions.
`crit`	Critical conditions.
`alert`	Immediate action required.
`emerg`	System is unstable.

All messages of the specified priority and higher are logged according to the specified action. An asterisk (`*`) wildcard specifies all facilities or priorities. Separate the names of multiple facilities and priorities on a line with commas (`,`). Separate multiple filters on one line with semicolons (;). Precede a priority with an exclamation mark (`!`) to select all messages except those with that priority.

The following are examples of facility/priority-based filters.

Select all kernel messages with any priority.

```
kern.*
```

Select all mail messages with `crit` or higher priority.

```
mail.crit
```

Select all `daemon` and `kern` messages with `warning` or `err` priority.

```
daemon,kern.warning,err
```

Select all `cron` messages except those with `info` or `debug` priority.

```
cron.!info,!debug
```

By default, `/etc/rsyslog.conf` includes the following rules:

```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                                 /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none               /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                             /var/log/secure

# Log all the mail messages in one place.
mail.*                                                 -/var/log/maillog

# Log cron stuff
cron.*                                                 /var/log/cron

# Everybody gets emergency messages
*.emerg                                                *

# Save news errors of level crit and higher in a special file.
uucp,news.crit                                         /var/log/spooler

# Save boot messages also to boot.log
local7.*                                               /var/log/boot.log
```

You can send the logs to a central log server over TCP by adding the following entry to the `forwarding rules` section of `/etc/rsyslog.conf` on each log client:

```
*.*         @@logsvr:port
```

where `logsvr` is the domain name or IP address of the log server and port is the port number (usually, 514).

On the log server, add the following entry to the `MODULES` section of `/etc/rsyslog.conf`:

```
$ModLoad imtcp
$InputTCPServerRun port
```

where `port` corresponds to the port number that you set on the log clients.

To manage the rotation and archival of the correct logs, edit `/etc/logrotate.d/syslog` so that it references each of the log files that are defined in the `RULES` section of `/etc/rsyslog.conf`. You can configure how often the logs are rotated and how many past copies of the logs are archived by editing `/etc/logrotate.conf`.

It is recommended that you configure Logwatch on your log server to monitor the logs for suspicious messages, and disable Logwatch on log clients. However, if you do use Logwatch, disable high precision timestamps by adding the following entry to the GLOBAL DIRECTIVES section of `/etc/rsyslog.conf` on each system:

```
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
```

For more information, see the `logrotate(8)`, `logwatch(8)`, `rsyslogd(8)` and `rsyslog.conf(5)` manual pages, the HTML documentation in the `/usr/share/doc/rsyslog-5.8.10` directory, and the documentation at http://www.rsyslog.com/doc/manual.html.

26.7.1 Configuring Logwatch

Logwatch is a monitoring system that you can configure to report on areas of interest in the system logs. After you install the `logwatch` package, the `/etc/cron.daily/0logwatch` script runs every night and sends an email report to `root`. You can set local configuration options in `/etc/logwatch/conf/logwatch.conf` that override the main configuration file `/usr/share/logwatch/default.conf/logwatch.conf`, including:

- Log files to monitor, including log files that are stored for other hosts.

- Names of services to monitor, or to be excluded from monitoring.

- Level of detail to report.

- User to be sent an emailed report.

You can also run `logwatch` directly from the command line.

For more information, see the `logwatch(8)` manual page.

26.8 About Process Accounting

The `psacct` package implements the process accounting service in addition to the following utilities that you can use to monitor process activities:

`ac`	Displays connection times in hours for a user as recorded in the `wtmp` file (by default, `/var/log/wtmp`).
`accton`	Turns on process accounting to the specified file. If you do not specify a file name argument, process accounting is stopped. The default system accounting file is `/var/account/pacct`.
`lastcomm`	Displays information about previously executed commands as recorded in the system accounting file.
`sa`	Summarizes information about previously executed commands as recorded in the system accounting file.

> **Note**
>
> As for any logging activity, ensure that the file system has enough space to store the system accounting and `wtmp` files. Monitor the size of the files and, if necessary, truncate them.

For more information, see the `ac(1)`, `accton(8)`, `lastcomm(1)`, and `sa(8)` manual pages.

26.9 Security Guidelines

The following sections provide guidelines that help secure your Oracle Linux system.

26.9.1 Minimizing the Software Footprint

On systems on which Oracle Linux has been installed, remove unneeded RPMs to minimize the software footprint. For example, you could uninstall the X Windows package (`xorg-x11-server-Xorg`) if it is not required on a server system.

To discover which package provides a given command or file, use the `yum provides` command as shown in the following example:

```
# yum provides /usr/sbin/sestatus
...
policycoreutils-2.0.83-19.24.0.1.el6.x86_64 : SELinux policy core utilities
Repo        : installed
Matched from:
Other       : Provides-match: /usr/sbin/sestatus
```

To display the files that a package provides, use the `repoquery` utility, which is included in the `yum-utils` package. For example, the following command lists the files that the `btrfs-progs` package provides.

```
# repoquery -l btrfs-progs
/sbin/btrfs
/sbin/btrfs-convert
/sbin/btrfs-debug-tree
.
.
.
```

To uninstall a package, use the `yum remove` command, as shown in this example:

```
# yum remove xinetd
Loaded plugins: refresh-packagekit, security
Setting up Remove Process
Resolving Dependencies
--> Running transaction check
---> Package xinetd.x86_64 2:2.3.14-35.el6_3 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package        Arch          Version                    Repository        Size
================================================================================
Removing:
 xinetd         x86_64        2:2.3.14-35.el6_3          @ol6_latest       259 k

Transaction Summary
================================================================================
Remove        1 Package(s)

Installed size: 259 k
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Erasing    : 2:xinetd-2.3.14-35.el6_3.x86_64                              1/1
  Verifying  : 2:xinetd-2.3.14-35.el6_3.x86_64                              1/1

Removed:
  xinetd.x86_64 2:2.3.14-35.el6_3

Complete!
```

The following table lists packages that you should not install or that you should remove using `yum remove` if they are already installed.

Package	Description
`krb5-appl-clients`	Kerberos versions of `ftp`, `rcp`, `rlogin`, `rsh` and `telnet`. If possible, use SSH instead.
`rsh`, `rsh-server`	`rcp`, `rlogin`, and `rsh` use unencrypted communication that can be snooped. Use SSH instead.
`samba`	Network services used by Samba. Remove this package if the system is not acting as an Active Directory server, a domain controller, or as a domain member, and it does not provide Microsoft Windows file and print sharing functionality.
`talk`, `talk-server`	`talk` is considered obsolete.
`telnet`, `telnet-server`	`telnet` uses unencrypted communication that can be snooped. Use SSH instead.
`tftp`, `tftp-server`	TFTP uses unencrypted communication that can be snooped. Use only if required to support legacy hardware. If possible, use SSH or other secure protocol instead.
`xinetd`	The security model used by the Internet listener daemon is deprecated.
`ypbind`, `ypserv`	The security model used by NIS is inherently flawed. Use an alternative such as LDAP or Kerberos instead.

26.9.2 Configuring System Logging

Verify that the system logging service `rsyslog` is running:

```
# systemctl status rsyslog
rsyslogd (pid  1632) is running...
```

If the service is not running, start it and enable it to start when the system is rebooted:

```
# systemctl start rsyslog
# systemctl enable rsyslog
```

Ensure that each log file referenced in `/etc/rsyslog.conf` exists and is owned and only readable by `root`:

```
# touch logfile
# chown root:root logfile
# chmod 0600 logfile
```

It is also recommended that you use a central log server and that you configure Logwatch on that server. See Section 26.7, "About System Logging".

26.9.3 Disabling Core Dumps

Core dumps can contain information that an attacker might be able to exploit and they take up a large amount of disk space. To prevent the system creating core dumps when the operating system terminates a program due to a segment violation or other unexpected error, add the following line to `/etc/security/limits.conf`:

```
*   hard   core   0
```

You can restrict access to core dumps to certain users or groups, as described in the `limits.conf(5)` manual page.

By default, the system prevents `setuid` and `setgid` programs, programs that have changed credentials, and programs whose binaries do not have read permission from dumping core. To ensure that the setting is permanently recorded, add the following lines to `/etc/sysctl.conf`:

```
# Disallow core dumping by setuid and setgid programs
fs.suid_dumpable = 0
```

and then run the `sysctl -p` command.

> **Note**
>
> A value of 1 permits core dumps that are readable by the owner of the dumping process. A value of 2 permits core dumps that are readable only by `root` for debugging purposes.

26.9.4 Minimizing Active Services

Restrict services to only those that a server requires. The default installation for an Oracle Linux server configures a minimal set of services:

- `cupsd` and `lpd` (print services)

- `sendmail` (email delivery service)

- `sshd` (openSSH services)

If possible, configure one type of service per physical machine, virtual machine, or Linux Container. This technique limits exposure if a system is compromised.

If a service is not used, remove the software packages that are associated with the service. If it is not possible to remove a service because of software dependencies, use the `systemctl` command to disable the service.

For services that are in use, apply the latest Oracle support patches and security updates to keep software packages up to date. To protect against unauthorized changes, ensure that the `/etc/services` file is owned by `root` and writable only by `root`.

```
# ls -Z /etc/services
-rw-r--r--. root root system_u:object_r:etc_t:SystemLow /etc/services
```

Unless specifically stated otherwise, consider disabling the services in the following table if they are not used on your system:

Service	Description
anacron	Executes commands periodically. Primarily intended for use on laptop and user desktop machines that do not run continuously.
automount	Manages mount points for the automatic file-system mounter. Disable this service on servers that do not require automounter functionality.
bluetooth	Supports the connections of Bluetooth devices. Primarily intended for use on laptop and user desktop machines. Bluetooth provides an additional potential attack surface. Disable this service on servers that do not require Bluetooth functionality.

Service	Description
gpm	(General Purpose Mouse) Provides support for the mouse pointer in a text console.
hidd	(Bluetooth Human Interface Device daemon) Provides support for Bluetooth input devices such as a keyboard or mouse. Primarily intended for use on laptop and user desktop machines. Bluetooth provides an additional potential attack surface. Disable this service on servers that do not require Bluetooth functionality.
irqbalance	Distributes hardware interrupts across processors on a multiprocessor system. Disable this service on servers that do not require this functionality.
iscsi	Controls logging in to iSCSI targets and scanning of iSCSI devices. Disable this service on servers that do not access iSCSI devices.
iscsid	Implements control and management for the iSCSI protocol. Disable this service on servers that do not access iSCSI devices.
kdump	Allows a kdump kernel to be loaded into memory at boot time or a kernel dump to be saved if the system panics. Disable this service on servers that you do not use for debugging or testing.
mcstrans	Controls the SELinux Context Translation System service.
mdmonitor	Checks the status of all software RAID arrays on the system. Disable this service on servers that do not use software RAID.
pcscd	(PC/SC Smart Card Daemon) Supports communication with smart-card readers. Primarily intended for use on laptop and user desktop machines to support smart-card authentication. Disable this service on servers that do not use smart-card authentication.
sandbox	Sets up /tmp, /var/tmp, and home directories to be used with the pam_namespace, sandbox, and xguest application confinement utilities. Disable this service if you do not use these programs.
setroubleshoot	Controls the SELinux Troubleshooting service, which provides information about SELinux Access Vector Cache (AVC) denials to the sealert tool.
smartd	Communicates with the Self-Monitoring, Analysis and Reporting Technology (SMART) systems that are integrated into many ATA-3 and later, and SCSI-3 disk drives. SMART systems monitor disk drives to measure reliability, predict disk degradation and failure, and perform drive testing.
xfs	Caches fonts in memory to improve the performance of X Window System applications.

You should consider disabling the following network services if they are not used on your system:

Service	Description
avahi-daemon	Implements Apple's Zero configuration networking (also known as Rendezvous or Bonjour). Primarily intended for use on laptop and user desktop machines to support music and file sharing. Disable this service on servers that do not require this functionality.
cups	Implements the Common UNIX Printing System. Disable this service on servers that do not need to provide this functionality.
hplip	Implements HP Linux Imaging and Printing to support faxing, printing, and scanning operations on HP inkjet and laser printers. Disable this service on servers that do not require this functionality.

Service	Description
`isdn`	(Integrated Services Digital Network) Provides support for network connections over ISDN devices. Disable this service on servers that do not directly control ISDN devices.
`netfs`	Mounts and unmounts network file systems, including NCP, NFS, and SMB. Disable this service on servers that do not require this functionality.
`network`	Activates all network interfaces that are configured to start at boot time.
`NetworkManager`	Switches network connections automatically to use the best connection that is available.
`nfslock`	Implements the Network Status Monitor (NSM) used by NFS. Disable this service on servers that do not require this functionality.
`nmb`	Provides NetBIOS name services used by Samba. Disable this service and remove the `samba` package if the system is not acting as an Active Directory server, a domain controller, or as a domain member, and it does not provide Microsoft Windows file and print sharing functionality.
`portmap`	Implements Remote Procedure Call (RPC) support for NFS. Disable this service on servers that do not require this functionality.
`rhnsd`	Queries the Unbreakable Linux Network (ULN) for updates and information.
`rpcgssd`	Used by NFS. Disable this service on servers that do not require this functionality.
`rpcidmapd`	Used by NFS. Disable this service on servers that do not require this functionality.
`smb`	Provides SMB network services used by Samba. Disable this service and remove the `samba` package if the system is not acting as an Active Directory server, a domain controller, or as a domain member, and it does not provide Microsoft Windows file and print sharing functionality.

To stop a service and prevent it from starting when you reboot the system, used the following commands:

```
# systemctl stop service_name
# systemctl disable service_name
```

26.9.5 Locking Down Network Services

> **Note**
>
> It is recommended that you do not install the `xinetd` Internet listener daemon. If you do not need this service, remove the package altogether by using the `yum remove xinetd` command.

If you must enable `xinetd` on your system, minimize the network services that `xinetd` can launch by disabling those services that are defined in the configuration files in `/etc/xinetd.d` and which are not needed.

To counter potential Denial of Service (DoS) attacks, you can configure the resource limits for such services by editing `/etc/xinetd.conf` and related configuration files. For example, you can set limits for the connection rate, the number of connection instances to a service, and the number of connections from an IP address:

```
# Maximum number of connections per second and
# number of seconds for which a service is disabled
```

```
# if the maximum number of connections is exceeded
cps            = 50 10

# Maximum number of connections to a service
instances      = 50

# Maximum number of connections from an IP address
per_source     = 10
```

For more information, see the `xinetd(8)` and `/etc/xinetd.conf(5)` manual pages.

26.9.6 Configuring a Packet-filtering Firewall

You can configure the Netfilter feature to act as a packet-filtering firewall that uses rules to determine whether network packets are received, dropped, or forwarded.

The primary interfaces for configuring the packet-filter rules are the `iptables` and `ip6tables` utilities and the Firewall Configuration Tool GUI (`firewall-config`). By default, the rules should drop any packets that are not destined for a service that the server hosts or that originate from networks other than those to which you want to allow access.

In addition, Netfilter provides Network Address Translation (NAT) to hide IP addresses behind a public IP address, and IP masquerading to alter IP header information for routed packets. You can also set rule-based packet logging and define a dedicated log file in `/etc/syslog.conf`.

For more information, see Section 26.3, "About Packet-filtering Firewalls".

26.9.7 Configuring TCP Wrappers

The TCP wrappers feature mediates requests from clients to services, and control access based on rules that you define in the `/etc/hosts.deny` and `/etc/hosts.allow` files. You can restrict and permit service access for specific hosts or whole networks. A common way of using TCP wrappers is to detect intrusion attempts. For example, if a known malicious host or network attempts to access a service, you can deny access and send a warning message about the event to a log file or to the system console.

For more information, see Section 26.4, "About TCP Wrappers".

26.9.8 Configuring Kernel Parameters

You can use several kernel parameters to counteract various kinds of attack.

`kernel.randomize_va_space` controls Address Space Layout Randomization (ASLR), which can help defeat certain types of buffer overflow attacks. A value of 0 disables ASLR, 1 randomizes the positions of the stack, virtual dynamic shared object (VDSO) page, and shared memory regions, and 2 randomizes the positions of the stack, VDSO page, shared memory regions, and the data segment. The default and recommended setting is 2.

`net.ipv4.conf.all.accept_source_route` controls the handling of source-routed packets, which might have been generated outside the local network. A value of 0 rejects such packets, and 1 accepts them. The default and recommended setting is 0.

`net.ipv4.conf.all.rp_filter` controls reversed-path filtering of received packets to counter IP address spoofing. A value of 0 disables source validation, 1 causes packets to be dropped if the routing table entry for their source address does not match the network interface on which they arrive, and 2 causes packets to be dropped if source validation by reversed path fails (see RFC 1812). The default setting is 0. A value of 2 can cause otherwise valid packets to be dropped if the local network topology is complex and RIP or static routes are used.

`net.ipv4.icmp_echo_ignore_broadcasts` controls whether ICMP broadcasts are ignored to protect against Smurf DoS attacks. A value of 1 ignores such broadcasts, and 0 accepts them. The default and recommended setting is 1.

`net.ipv4.icmp_ignore_bogus_error_message` controls whether ICMP bogus error message responses are ignored. A value of 1 ignores such messages, and 0 accepts them. The default and recommended setting is 1.

To change the value of a kernel parameter, add the setting to `/etc/sysctl.conf`, for example:

```
kernel.randomize_va_space = 1
```

and then run the `sysctl -p` command.

26.9.9 Restricting Access to SSH Connections

The Secure Shell (SSH) allows protected, encrypted communication with other systems. As SSH is an entry point into the system, disable it if it is not required, or alternatively, edit the `/etc/ssh/sshd_config` file to restrict its use.

For example, the following setting does not allow `root` to log in using SSH:

```
PermitRootLogin no
```

You can restrict remote access to certain users and groups by specifying the `AllowUsers`, `AllowGroups`, `DenyUsers`, and `DenyGroups` settings, for example:

```
DenyUsers carol dan
AllowUsers alice bob
```

The `ClientAliveInterval` and `ClientAliveCountMax` settings cause the SSH client to time out automatically after a period of inactivity, for example:

```
# Disconnect client after 300 seconds of inactivity
ClientAliveCountMax 0
ClientAliveInterval 300
```

After making changes to the configuration file, restart the `sshd` service for your changes to take effect.

For more information, see the `sshd_config(5)` manual page.

26.9.10 Configuring File System Mounts, File Permissions, and File Ownerships

Use separate disk partitions for operating system and user data to prevent a *file system full* issue from impacting the operation of a server. For example, you might create separate partitions for `/home`, `/tmp`, p, `/oracle`, and so on.

Establish disk quotas to prevent a user from accidentally or intentionally filling up a file system and denying access to other users.

To prevent the operating system files and utilities from being altered during an attack, mount the `/usr` file system read-only. If you need to update any RPMs on the file system, use the `-o remount,rw` option with the `mount` command to remount `/usr` for both read and write access. After performing the update, use the `-o remount,ro` option to return the `/usr` file system to read-only mode.

To limit user access to non-`root` local file systems such as `/tmp` or removable storage partitions, specify the `-o noexec, nosuid, nodev` options to `mount`. These option prevent the execution of binaries (but not scripts), prevent the `setuid` bit from having any effect, and prevent the use of device files.

Use the `find` command to check for unowned files and directories on each file system, for example:

```
# find mount_point -mount -type f -nouser -o -nogroup -exec ls -l {} \;
# find mount_point -mount -type d -nouser -o -nogroup -exec ls -l {} \;
```

Unowned files and directories might be associated with a deleted user account, they might indicate an error with software installation or deleting, or they might a sign of an intrusion on the system. Correct the permissions and ownership of the files and directories that you find, or remove them. If possible, investigate and correct the problem that led to their creation.

Use the `find` command to check for world-writable directories on each file system, for example:

```
# find mount_point -mount -type d -perm /o+w -exec ls -l {} \;
```

Investigate any world-writable directory that is owned by a user other than a system user. The user can remove or change any file that other users write to the directory. Correct the permissions and ownership of the directories that you find, or remove them.

You can also use `find` to check for `setuid` and `setgid` executables.

```
# find path -type f \( -perm -4000 -o -perm -2000 \) -exec ls -l {} \;
```

If the `setuid` and `setgid` bits are set, an executable can perform a task that requires other rights, such as `root` privileges. However, buffer overrun attacks can exploit such executables to run unauthorized code with the rights of the exploited process.

If you want to stop a `setuid` and `setgid` executable from being used by non-`root` users, you can use the following commands to unset the `setuid` or `setgid` bit:

```
# chmod u-s file
# chmod g-s file
```

The following table lists programs for which you might want to consider unsetting the `setuid` and `setgid`:

Program File	Bit Set	Description of Usage
`/usr/bin/chage`	`setuid`	Finds out password aging information (via the `-l` option).
`/usr/bin/chfn`	`setuid`	Changes `finger` information.
`/usr/bin/chsh`	`setuid`	Changes the login shell.
`/usr/bin/crontab`	`setuid`	Edits, lists, or removes a `crontab` file.
`/usr/bin/wall`	`setgid`	Sends a system-wide message.
`/usr/bin/write`	`setgid`	Sends a message to another user.
`/usr/bin/Xorg`	`setuid`	Invokes the X Windows server.
`/usr/libexec/openssh/ ssh-keysign`	`setuid`	Runs the SSH helper program for host-based authentication.
`/usr/sbin/mount.nfs`	`setuid`	Mounts an NFS file system.
		Note `/sbin/mount.nfs4`, `/sbin/umount.nfs`, and `/sbin/umount.nfs4` are symbolic links to this file.
`/usr/sbin/netreport`	`setgid`	Requests notification of changes to network interfaces.

Program File	Bit Set	Description of Usage
`/usr/sbin/usernetctl`	`setuid`	Controls network interfaces. Permission for a user to alter the state of a network interface also requires `USERCTL=yes` to be set in the interface file. You can also grant users and groups the privilege to run the `ip` command by creating a suitable entry in the `/etc/sudoers` file.

Note

This list is not exhaustive as many optional packages contain `setuid` and `setgid` programs.

26.9.11 Checking User Accounts and Privileges

Check the system for unlocked user accounts on a regular basis, for example using a command such as the following:

```
# for u in `cat /etc/passwd | cut -d: -f1 | sort`; do passwd -S $u; done
abrt LK 2012-06-28 0 99999 7 -1 (Password locked.)
adm LK 2011-10-13 0 99999 7 -1 (Alternate authentication scheme in use.)
apache LK 2012-06-28 0 99999 7 -1 (Password locked.)
avahi LK 2012-06-28 0 99999 7 -1 (Password locked.)
avahi-autoipd LK 2012-06-28 0 99999 7 -1 (Password locked.)
bin LK 2011-10-13 0 99999 7 -1 (Alternate authentication scheme in use.)
...
```

In the output from this command, the second field shows if a user account is locked (`LK`), does not have a password (`NP`), or has a valid password (`PS`). The third field shows the date on which the user last changed their password. The remaining fields show the minimum age, maximum age, warning period, and inactivity period for the password and additional information about the password's status. The unit of time is days.

Use the `passwd` command to set passwords on any accounts that are not protected.

Use `passwd -l` to lock unused accounts. Alternatively, use `userdel` to remove the accounts entirely.

For more information, see the `passwd(1)` and `userdel(8)` manual pages.

To specify how users' passwords are aged, edit the following settings in the `/etc/login.defs` file:

Setting	Description
`PASS_MAX_DAYS`	Maximum number of days for which a password can be used before it must be changed. The default value is 99,999 days.
`PASS_MIN_DAYS`	Minimum number of days that is allowed between password changes. The default value is 0 days.
`PASS_WARN_AGE`	Number of days warning that is given before a password expires. The default value is 7 days.

For more information, see the `login.defs(5)` manual page.

To change how long a user's account can be inactive before it is locked, use the `usermod` command. For example, to set the inactivity period to 30 days:

```
# usermod -f 30 username
```

To change the default inactivity period for new user accounts, use the `useradd` command:

```
# useradd -D -f 30
```

A value of -1 specifies that user accounts are not locked due to inactivity.

For more information, see the `useradd(8)` and `usermod(8)` manual pages.

Verify that no user accounts other than `root` have a user ID of 0.

```
# awk -F":" '$3 == 0 { print $1 }' /etc/passwd
root
```

If you install software that creates a default user account and password, change the vendor's default password immediately. Centralized user authentication using an LDAP implementation such as OpenLDAP can help to simplify user authentication and management tasks, and also reduces the risk arising from unused accounts or accounts without a password.

By default, an Oracle Linux system is configured so that you cannot log in directly as `root`. You must log in as a named user before using either `su` or `sudo` to perform tasks as `root`. This configuration allows system accounting to trace the original login name of any user who performs a privileged administrative action. If you want to grant certain users authority to be able to perform specific administrative tasks via `sudo`, use the `visudo` command to modify the `/etc/sudoers` file. For example, the following entry grants the user `erin` the same privileges as `root` when using `sudo`, but defines a limited set of privileges to `frank` so that he can run commands such as `systemctl`, `rpm`, and `yum`:

```
erin            ALL=(ALL)       ALL
frank           ALL= SERVICES, SOFTWARE
```

26.9.11.1 Configuring User Authentication and Password Policies

The Pluggable Authentication Modules (PAM) feature allows you to enforce strong user authentication and password policies, including rules for password complexity, length, age, expiration and the reuse of previous passwords. You can configure PAM to block user access after too many failed login attempts, after normal working hours, or if too many concurrent sessions are opened.

PAM is highly customizable by its use of different modules with customisable parameters. For example, the default password integrity checking module `pam_pwquality.so` tests password strength. The PAM configuration file (`/etc/pam.d/system-auth`) contains the following default entries for testing a password's strength:

```
password  requisite   pam_pwquality.so try_first_pass local_users_only retry=3 authtok_type=
password  sufficient  pam_unix.so sha512 shadow nullok try_first_pass use_authtok
password  required    pam_deny.so
```

The line for `pam_pwquality.so` defines that a user gets three attempts to choose a good password. From the module's default settings, the password length must a minimum of six characters, of which three characters must be different from the previous password. The module only tests the quality of passwords for users who are defined in `/etc/passwd`.

The line for `pam_unix.so` specifies that the module tests the password previously specified in the stack before prompting for a password if necessary (`pam_pwquality` will already have performed such checks for users defined in `/etc/passwd`), uses SHA-512 password hashing and the `/etc/shadow` file, and allows access if the existing password is null.

You can modify the control flags and module parameters to change the checking that is performed when a user changes his or her password, for example:

```
password  required   pam_pwquality.so retry=3 minlen=8 difok=5 minclass=-1
password  required   pam_unix.so use_authtok sha512 shadow remember=5
password  required   pam_deny.so
```

The line for `pam_pwquality.so` defines that a user gets three attempts to choose a good password with a minimum of eight characters, of which five characters must be different from the previous password, and

which must contain at least one upper case letter, one lower case letter, one numeric digit, and one non-alphanumeric character.

The line for `pam_unix.so` specifies that the module does not perform password checking, uses SHA-512 password hashing and the `/etc/shadow` file, and saves information about the previous five passwords for each user in the `/etc/security/opasswd` file. As `nullok` is not specified, a user cannot change his or her password if the existing password is null.

The omission of the `try_first_pass` keyword means that the user is always asked for their existing password, even if he or she entered it for the same module or for a previous module in the stack.

For more information, see Section 24.7, "About Pluggable Authentication Modules" and the `pam_deny(8)`, `pam_pwquality(8)`, and `pam_unix(8)` manual pages.

An alternate way of defining password requirements is available by selecting the Password Options tab in the Authentication Configuration GUI (`system-config-authentication`).

Figure 26.2 shows the Authentication Configuration GUI with the Password Options tab selected.

Figure 26.2 Password Options

You can specify the minimum password length, minimum number of required character classes, which character classes are required, and the maximum number of consecutive characters and consecutive characters from the same class that are permitted.

Chapter 27 OpenSSH Configuration

Table of Contents

This chapter describes how to configure OpenSSH to support secure communication between networked systems.

27.1 About OpenSSH

OpenSSH is suite of network connectivity tools that provides secure communications between systems, including:

`scp`   Secure file copying.

`sftp`   Secure File Transfer Protocol (FTP).

`ssh`   Secure shell to log on to or run a command on a remote system.

`sshd`   Daemon that supports the OpenSSH services.

`ssh-keygen`   Creates DSA or RSA authentication keys.

Unlike utilities such as `rcp`, `ftp`, `telnet`, `rsh`, and `rlogin`, the OpenSSH tools encrypt all network packets between the client and server, including password authentication.

OpenSSH supports SSH protocol version 1 (SSH1) and version 2 (SSH2). In addition, OpenSSH provides a secure way of using graphical applications over a network by using X11 forwarding. It also provides a way to secure otherwise insecure TCP/IP protocols by using port forwarding.

27.2 OpenSSH Configuration Files

The following OpenSSH global configuration files are located in `/etc/ssh`:

`moduli`   Contains key-exchange information that is used to set up a secure connection.

`ssh_config`   Contains default client configuration settings that can be overridden by the settings in a user's `~/.ssh/config` file.

`ssh_host_dsa_key`   Contains the DSA private key for SSH2.

`ssh_host_dsa_key.pub`   Contains the DSA public key for SSH2.

`ssh_host_key`	Contains the RSA private key for SSH1.
`ssh_host_key.pub`	Contains the RSA public key for SSH1.
`ssh_host_rsa_key`	Contains the RSA private key for SSH2.
`ssh_host_rsa_key.pub`	Contains the RSA public key for SSH2.
`sshd_config`	Contains configuration settings for `sshd`.

Other files can be configured in this directory. For details, see the `sshd(8)` manual page.

For more information, see the `ssh_config(5)`, `sshd(8)`, and `sshd_config(5)` manual pages.

27.2.1 OpenSSH User Configuration Files

To use the OpenSSH tools, a user must have an account on both the client and server systems. The accounts do not need to be configured identically on each system.

User configuration files are located in the `.ssh` directory in a user's home directory (`~/.ssh`) on both the client and server. OpenSSH creates this directory and the `known_hosts` file when the user first uses an OpenSSH utility to connect to a remote system.

27.2.1.1 User Configuration Files in ~/.ssh on the Client

On the client side, the `~/.ssh/known_hosts` file contains the public host keys that OpenSSH has obtained from SSH servers. OpenSSH adds an entry for each new server to which a user connects.

In addition, the `~/.ssh` directory usually contains one of the following pairs of key files:

`id_dsa` and `id_dsa.pub`	Contain a user's SSH2 DSA private and public keys.
`id_rsa` and `id_rsa.pub`	Contains a user's SSH2 RSA private and public keys. SSH2 RSA is most commonly used key-pair type.
`identity` and `identity.pub`	Contains a user's SSH1 RSA private and public keys.

> **Caution**
>
> The private key file can be readable and writable by the user but must not be accessible to other users.

The optional `config` file contains client configuration settings.

> **Caution**
>
> A `config` file can be readable and writable by the user but must not be accessible to other users.

For more information, see the `ssh(1)` and `ssh-keygen(1)` manual pages.

27.2.1.2 User Configuration Files in ~/.ssh on the Server

On the server side, the `~/.ssh` directory usually contains the following files:

`authorized_keys`	Contains your authorized public keys. The server uses the signed public key in this file to authenticate a client.

config   Contains client configuration settings. This file is optional.

> **Caution**
>
> A `config` file can be readable and writable by the user but must not be accessible to other users.

environment   Contains definitions of environment variables. This file is optional.

rc   Contains commands that `ssh` executes when a user logs in, before the user's shell or command runs. This file is optional.

For more information, see the `ssh(1)` and `ssh_config(5)` manual pages.

27.3 Configuring an OpenSSH Server

> **Note**
>
> The default Oracle Linux installation includes the `openssh` and `openssh-server` packages, but does not enable the `sshd` service.

To configure an OpenSSH server:

1. Install or update the `openssh` and `openssh-server` packages:

   ```
   # yum install openssh openssh-server
   ```

2. Start the `sshd` service and configure it to start following a system reboot:

   ```
   # systemctl start sshd
   # systemctl enable sshd
   ```

You can set `sshd` configuration options for features such as Kerberos authentication, X11 forwarding, and port forwarding in the `/etc/ssh/sshd_config` file.

For more information, see the `sshd(8)` and `sshd_config(5)` manual pages.

27.4 Installing the OpenSSH Client Packages

> **Note**
>
> The default Oracle Linux installation includes the `openssh` and `openssh-client` packages.

To configure an OpenSSH client, install or update the `openssh` and `openssh-client` packages:

```
# yum install openssh openssh-client
```

27.5 Using the OpenSSH Utilities

By default, each time you use the OpenSSH utilities to connect to a remote system, you must provide your user name and password to the remote system. When you connect to an OpenSSH server for the first time, the OpenSSH client prompts you to confirm that you are connected to the correct system. In the following example, the `ssh` command is used to connect to the remote host `host04`:

```
$ ssh host04
```

```
The authenticity of host 'host04 (192.0.2.104)' can't be
established.
RSA key fingerprint is 65:ad:38:b2:8a:6c:69:f4:83:dd:3f:8f:ba:b4:85:c7.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'host04,192.0.2.104' (RSA) to the
list of known hosts.
```

When you enter `yes` to accept the connection to the server, the client adds the server's public host key to the your `~/.ssh/known_hosts` file. When you next connect to the remote server, the client compares the key in this file to the one that the server supplies. If the keys do not match, you see a warning such as the following:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: POSSIBLE DNS SPOOFING DETECTED!         @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
The RSA host key for host has changed,
and the key for the according IP address IP_address
is unchanged. This could either mean that
DNS SPOOFING is happening or the IP address for the host
and its host key have changed at the same time.
Offending key for IP in /home/user/.ssh/known_hosts:10
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is fingerprint
Please contact your system administrator.
Add correct host key in /home/user/.ssh/known_hosts to get rid of this message.
Offending key in /home/user/.ssh/known_hosts:53
RSA host key for host has changed and you have requested strict checking.
Host key verification failed.
```

Unless there is a reason for the remote server's host key to have changed, such as an upgrade of either the SSH software or the server, you should not try to connect to that machine until you have contacted its administrator about the situation.

27.5.1 Using ssh to Connect to Another System

The ssh command allows you to log in to a remote system, or to execute a command on a remote system:

```
$ ssh [options] [user@]host [command]
```

host is the name of the remote OpenSSH server to which you want to connect.

For example, to log in to `host04` with the same user name as on the local system, enter:

```
$ ssh host04
```

The remote system prompts you for your password on that system.

To connect as a different user, specify the user name and `@` symbol before the remote host name, for example:

```
$ ssh joe@host04
```

To execute a command on the remote system, specify the command as an argument, for example:

```
$ ssh joe@host04 ls ~/.ssh
```

`ssh` logs you in, executes the command, and then closes the connection.

For more information, see the `ssh(1)` manual page.

27.5.2 Using scp and sftp to Copy Files Between Systems

The `scp` command allows you to copy files or directories between systems. `scp` establishes a connection, copies the files, and then closes the connection.

To upload a local file to a remote system:

```
$ scp [options] local_file [user@]host[:remote_file]
```

For example, copy `testfile` to your home directory on `host04`:

```
$ scp testfile host04
```

Copy `testfile` to the same directory but change its name to `new_testfile`:

```
$ scp testfile host04:new_testfile
```

To download a file from a remote system to the local system:

```
$ scp [options] [user@]host[:remote_file] local_file
```

The `-r` option allows you to recursively copy the contents of directories. For example, copy the directory `remdir` and its contents from your home directory on remote `host04` to your local home directory:

```
$ scp -r host04:~/remdir ~
```

The `sftp` command is a secure alternative to `ftp` for file transfer between systems. Unlike `scp`, `sftp` allows you to browse the file system on the remote server before you copy any files.

To open an FTP connection to a remote system over SSH:

```
$ sftp [options] [user@]host
```

For example:

```
$ sftp host04
Connecting to host04...
guest@host04's password: password
sftp>
```

Enter `sftp` commands at the `sftp>` prompt. For example, use **put** to upload the file `newfile` from the local system to the remote system and **ls** to list it:

```
sftp> put newfile
Uploading newfile to /home/guest/newfile
foo                                          100% 1198     1.2KB/s   00:01
sftp> ls foo
foo
```

Enter **help** or **?** to display a list of available commands. Enter **bye**, **exit**, or **quit** to close the connection and exit `sftp`.

For more information, see the `ssh(1)` and `sftp(1)` manual pages.

27.5.3 Using ssh-keygen to Generate Pairs of Authentication Keys

The `ssh-keygen` command generate a public and private authentication key pair. Such authentication keys allow you to connect to a remote system without needing to supply a password each time that you connect. Each user must generate their own pair of keys. If `root` generates key pairs, only `root` can use those keys.

To create a public and private SSH2 RSA key pair:

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/guest/.ssh/id_rsa): <Enter>
Created directory '/home/guest/.ssh'.
Enter passphrase (empty for no passphrase): password
Enter same passphrase again: password
Your identification has been saved in /home/guest/.ssh/id_rsa.
Your public key has been saved in /home/guest/.ssh/id_rsa.pub.
The key fingerprint is:
5e:d2:66:f4:2c:c5:cc:07:92:97:c9:30:0b:11:90:59 guest@host01
The key's randomart image is:
+--[ RSA 2048]----+
|      .=Eo++.o    |
|       o  ..B=.   |
|          o.=  .  |
|          o + .   |
|         S * o    |
|          . = .   |
|             .    |
|           .      |
|                  |
+-----------------+
```

To generate an SSH1 RSA or SSH2 DSA key pair, specify the `-t rsa1` or `-t dsa` options.

For security, in case an attacker gains access to your private key, you can specify an passphrase to encrypt your private key. If you encrypt your private key, you must enter this passphrase each time that you use the key. If you do not specify a passphrase, you are not prompted.

`ssh-keygen` generates a private key file and a public key file in `~/.ssh` (unless you specify an alternate directory for the private key file):

```
$ ls -l ~/.ssh
total 8
-rw-------. 1 guest guest 1743 Apr 13 12:07 id_rsa
-rw-r--r--. 1 guest guest  397 Apr 13 12:07 id_rsa.pub
```

For more information, see the `ssh-keygen(1)` manual page.

27.5.4 Enabling Remote System Access Without Requiring a Password

To be able to use the OpenSSH utilities to access a remote system without supplying a password each time that you connect:

1. Use `ssh-keygen` to generate a public and private key pair, for example:

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa): <Enter>
Created directory '/home/user/.ssh'.
Enter passphrase (empty for no passphrase): <Enter>
Enter same passphrase again: <Enter>
```

```
...
```

Press `Enter` each time that the command prompts you to enter a passphrase.

2. Use the `ssh-copy-id` script to append the public key in the local `~/.ssh/id_rsa.pub` file to the `~/.ssh/authorized_keys` file on the remote system, for example:

```
$ ssh-copy-id remote_user@host
remote_user@host's password: remote_password
Now try logging into the machine, with "ssh 'remote_user@host'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.
```

When prompted, enter your password for the remote system.

The script also changes the permissions of `~/.ssh` and `~/.ssh/authorized_keys` on the remote system to disallow access by your group.

You can now use the OpenSSH utilities to access the remote system without supplying a password. As the script suggests, you should use `ssh` to log into the remote system to verify that the `~/.ssh/authorized_keys` file contains only the keys for the systems from which you expect to connect. For example:

```
$ ssh remote_user@host
Last login: Thu Jun 13 08:33:58 2013 from local_host
host$ cat .ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA6OabJhWABsZ4F3mcjEPT3sxnXx1OoUcvuCiM6fg5s/ER
... FF488hBOk2ebpo38fHPPK1/rsOEKX9Kp9QWH+IfASI8q09xQ== local_user@local_host
host$ logout
Connection to host closed.
$
```

3. Verify that the permissions on the remote `~/.ssh` directory and `~/.ssh/authorized_keys` file allow access only by you:

```
$ ssh remote_user@host ls -al .ssh
total 4
drwx------+ 2 remote_user group   5 Jun 12 08:33 .
drwxr-xr-x+ 3 remote_user group   9 Jun 12 08:32 ..
-rw-------+ 1 remote_user group 397 Jun 12 08:33 authorized_keys
$ ssh remote_user@host getfacl .ssh
# file: .ssh
# owner: remote_user
# group: group
user::rwx
group::---
mask::rwx
other::---

$ ssh remote_user@host getfacl .ssh/authorized_keys
# file: .ssh/authorized_keys
# owner: remote_user
# group: group
user::rw-
group::---
mask::rwx
other::---
```

If necessary, correct the permissions:

```
$ ssh remote_user@host 'umask 077; /sbin/restorecon .ssh'
$ ssh remote_user@host 'umask 077; /sbin/restorecon .ssh/authorized_keys'
```

> **Note**
>
> If your user names are the same on the client and the server systems, you do not need to specify your remote user name and the `@` symbol.

4. If your user names are different on the client and the server systems, create a `~/.ssh/config` file with permissions 600 on the remote system that defines your local user name, for example:

```
$ ssh remote_user@host echo -e "Host *\\\nUser local_user" '>>' .ssh/config
$ ssh remote_user@host cat .ssh/config
Host *
User local_user
$ ssh remote_user@host 'umask 077; /sbin/restorecon .ssh/config'
```

You should now be able to access the remote system without needing to specify your remote user name, for example:

```
$ ssh host ls -l .ssh/config
-rw-------+ 1 remote_user group 37 Jun 12 08:34 .ssh/config
$ ssh host getfacl .ssh/config
# file: .ssh/config
# owner: remote_user
# group: group
user::rw-
group::---
mask::rwx
other::---
```

For more information, see the `ssh-copy-id(1)`, `ssh-keygen(1)`, and `ssh_config(5)` manual pages.

Part V Containers

This section contains the following chapters:

- Chapter 28, *Linux Containers* describes how to use Linux Containers (LXC) to isolate applications and entire operating system images from the other processes that are running on a host system.

> **Note**
>
> Information on using the Docker engine to manage containers and images under Oracle Linux is provided in the *Oracle Linux Docker User's Guide* available at http://docs.oracle.com/cd/E52668_01/E75728/html/.

Table of Contents

Chapter 28 Linux Containers

Table of Contents

This chapter describes how to use Linux Containers (LXC) to isolate applications and entire operating system images from the other processes that are running on a host system. The version of LXC described here is 1.0.7 or later, which has some significant enhancements over previous versions.

For information about how to use the Docker Engine to create application containers, see the *Oracle Linux Docker User's Guide*.

28.1 About Linux Containers

> **Note**
>
> Prior to UEK R3, LXC was a Technology Preview feature that was made available for testing and evaluation purposes, but was not recommended for production systems. LXC is a supported feature with UEK R3 and UEK R4.

The Linux Containers (LXC) feature is a lightweight virtualization mechanism that does not require you to set up a virtual machine on an emulation of physical hardware. The Linux Containers feature takes the cgroups resource management facilities as its basis and adds POSIX file capabilities to implement process and network isolation. You can run a single application within a container (an *application container*) whose name space is isolated from the other processes on the system in a similar manner to a `chroot` jail. However, the main use of Linux Containers is to allow you to run a complete copy of the Linux operating system in a container (a *system container*) without the overhead of running a level-2 hypervisor such as VirtualBox. In fact, the container is sharing the kernel with the host system, so its processes and file system are completely visible from the host. When you are logged into the container, you only see its file system and process space. Because the kernel is shared, you are limited to the modules and drivers that it has loaded.

Typical use cases for Linux Containers are:

- Running Oracle Linux 5, Oracle Linux 6, and Oracle Linux 7 containers in parallel. You can run an Oracle Linux 5 container on an Oracle Linux 7 system with the UEK R3 or UEKR4 kernel, even though UEK R3 and UEK R4 are not supported for Oracle Linux 5. You can also run an i386 container on an x86_64 kernel. For more information, see Section 28.1.1, "Supported Oracle Linux Container Versions".

- Running applications that are supported only by Oracle Linux 5 in an Oracle Linux 5 container on an Oracle Linux 7 host. However, incompatibilities might exist in the modules and drivers that are available.

- Running many copies of application configurations on the same system. An example configuration would be a LAMP stack, which combines Linux, Apache HTTP server, MySQL, and Perl, PHP, or Python scripts to provide specialised web services.

- Creating sandbox environments for development and testing.

- Providing user environments whose resources can be tightly controlled, but which do not require the hardware resources of full virtualization solutions.

- Creating containers where each container appears to have its own IP address. For example you can use the `lxc-sshd` template script to create isolated environments for untrusted users. Each container runs an `sshd` daemon to handle logins. By bridging a container's Virtual Ethernet interface to the host's network interface, each container can appear to have its own IP address on a LAN.

When you use the `lxc-start` command to start a system container, by default the copy of `/sbin/init` (for an Oracle Linux 6 or earlier container) or `/usr/lib/systemd/systemd` (for an Oracle Linux 7 container) in the container is started to spawn other processes in the container's process space. Any system calls or device access are handled by the kernel running on the host. If you need to run different kernel versions or different operating systems from the host, use a full virtualization solution such as Oracle VM or Oracle VM VirtualBox instead of Linux Containers.

There are a number of configuration steps that you need to perform on the file system image for a container so that it can run correctly:

- Disable any `init` or `systemd` scripts that load modules to access hardware directly.

- Disable `udev` and instead create static device nodes in `/dev` for any hardware that needs to be accessible from within the container.

- Configure the network interface so that it is bridged to the network interface of the host system.

LXC provides a number of template scripts in `/usr/share/lxc/templates` that perform much of the required configuration of system containers for you. However, it is likely that you will need to modify the script to allow the container to work correctly as the scripts cannot anticipate the idiosyncrasies of your system's configuration. You use the `lxc-create` command to create a system container by invoking a template script. For example, the `lxc-busybox` template script creates a lightweight BusyBox system container.

The example system container in this chapter uses the template script for Oracle Linux (`lxc-oracle`). The container is created on a btrfs file system (`/container`) to take advantage of its snapshot feature. A btrfs file system allows you to create a subvolume that contains the root file system (`rootfs`) of a container, and to quickly create new containers by cloning this subvolume.

You can use control groups to limit the system resources that are available to applications such as web servers or databases that are running in the container.

Application containers are not created by using template scripts. Instead, an application container mounts all or part of the host's root file system to provide access to the binaries and libraries that the application requires. You use the `lxc-execute` command to invoke `/usr/sbin/init.lxc` (a cut-down version of

`/sbin/init`) in the container. `init.lxc` mounts any required directories such as `/proc`, `/dev/shm`, and `/dev/mqueue`, executes the specified application program, and then waits for it to finish executing. When the application exits, the container instance ceases to exist.

28.1.1 Supported Oracle Linux Container Versions

All versions of Oracle Linux 7, running `kernel-uek-3.8.13-35.3.1` or later, support the following container versions:

- Oracle Linux 5.9 or later

- Oracle Linux 6.5 or later

- Oracle Linux 7.0 or later

Note that subsequent versions of Oracle Linux 7 and UEK are tested to support the listed container versions. Exceptions, if any, are listed in the release notes for the version of Oracle Linux 7 affected.

28.2 Configuring Operating System Containers

The procedures in the following sections describe how to set up Linux Containers that contain a copy of the root file system installed from packages on the Oracle Linux Yum Server.

- Section 28.2.1, "Installing and Configuring the Software"

- Section 28.2.2, "Setting up the File System for the Containers"

- Section 28.2.3, "Creating and Starting a Container"

> **Note**
>
> Throughout the following sections in this chapter, the prompts `[root@host ~]#` and `[root@ol6ctr1 ~]#` distinguish between commands run by `root` on the host and in the container.

28.2.1 Installing and Configuring the Software

To install and configure the software that is required to run Linux Containers:

1. Use `yum` to install the `btrfs-progs` package.

   ```
   [root@host ~]# yum install btrfs-progs
   ```

2. Install the `lxc` and `wget` packages.

   ```
   [root@host ~]# yum install lxc wget
   ```

   This command installs all of the required packages, such as `libvirt` and `lxc-libs`. The LXC template scripts are installed in `/usr/share/lxc/templates`. LXC uses the virtualization management service to support network bridging for containers. LXC uses `wget` to download packages from the Oracle Linux Yum Server.

3. Start the virtualization management service, `libvirtd`, and configure the service to start at boot time.

   ```
   [root@host ~]# systemctl start libvirtd
   [root@host ~]# systemctl enable libvirtd
   ```

   LXC uses the virtualization management service to support network bridging for containers.

4. If you are going to compile applications that require the LXC header files and libraries, install the `lxc-devel` package.

```
[root@host ~]# yum install lxc-devel
```

28.2.2 Setting up the File System for the Containers

Note

The LXC template scripts assume that containers are created in `/container`. You must edit the script if your system's configuration differs from this assumption.

To set up the `/container` file system:

1. Create a btrfs file system on a suitably sized device such as `/dev/sdb`, and create the `/container` mount point.

```
[root@host ~]# mkfs.btrfs /dev/sdb
[root@host ~]# mkdir /container
```

2. Mount the `/container` file system.

```
[root@host ~]# mount /dev/sdb /container
```

3. Add an entry for `/container` to the `/etc/fstab` file.

```
/dev/sdb        /container     btrfs     defaults    0 0
```

For more information, see Section 21.2, "About the Btrfs File System".

28.2.3 Creating and Starting a Container

Note

The procedure in this section uses the LXC template script for Oracle Linux (`lxc-oracle`), which is located in `/usr/share/lxc/templates`.

An Oracle Linux container requires a minimum of 400 MB of disk space.

To create and start a container:

1. Create an Oracle Linux 6 container named `ol6ctr1` using the `lxc-oracle` template script.

```
[root@host ~]# lxc-create -n ol6ctr1 -B btrfs -t oracle -- --release=6.latest
Host is OracleEverything 7.0
Create configuration file /container/ol6ctr1/config
Yum installing release 6.latest for x86_64
  .
  .
  .
  yum-metadata-parser.x86_64 0:1.1.2-16.el6
  zlib.x86_64 0:1.2.3-29.el6

Complete!
Rebuilding rpm database
Patching container rootfs /container/ol6ctr1/rootfs for Oracle Linux 6.5
Configuring container for Oracle Linux 6.5
Added container user:oracle password:oracle
Added container user:root password:root
Container : /container/ol6ctr1/rootfs
Config    : /container/ol6ctr1/config
Network   : eth0 (veth) on virbr0
```

Note

For LXC version 1.0 and later, you must specify the `-B btrfs` option if you want to use the snapshot features of btrfs. For more information, see the `lxc-create(1)` manual page.

The `lxc-create` command runs the template script `lxc-oracle` to create the container in `/container/ol6ctr1` with the btrfs subvolume `/container/ol6ctr1/rootfs` as its root file system. The command then uses `yum` to install the latest available update of Oracle Linux 6 from the Oracle Linux Yum Server. It also writes the container's configuration settings to the file `/container/ol6ctr1/config` and its `fstab` file to `/container/ol6ctr1/fstab`. The default log file for the container is `/container/ol6ctr1/ol6ctr1.log`.

You can specify the following template options after the `--` option to `lxc-create`:

`-a`|`--arch=i386`|`x86_64`	Specifies the architecture. The default value is the architecture of the host.
`--baseurl=`*pkg_repo*	Specify the file URI of a package repository. You must also use the `--arch` and `--release` options to specify the architecture and the release, for example:

```
# mount -o loop OracleLinux-R7-GA-Everything-x86_64-dvd.iso /mnt
# lxc-create -n ol70beta -B btrfs -t oracle -- -R 7.0 -a x86_64 \
  --baseurl=file:///mnt/Server
```

`-P`|`--patch=`*path*	Patch the `rootfs` at the specified path.
`--privileged`[`=rt`]	Allows you to adjust the values of certain kernel parameters under the `/proc` hierarchy. The container uses a privilege configuration file, which mounts `/proc` read-only with some exceptions. See Section 28.9, "Configuring Kernel Parameter Settings for Oracle Linux Containers". This option also enables the `CAP_SYS_NICE` capability, which allows you to set negative `nice` values (that is, more favored for scheduling) for processes from within the container. If you specify the `=rt` (real-time) modifier, you can configure the `lxc.cgroup.cpu.rt_runtime_us` setting in the container's configuration file or when you start the container. This setting specifies the maximum continuous period in microseconds for which the container has access to CPU resources from the base period set by the system-wide value of `cpu.rt_period_us`. Otherwise, a container uses the system-wide value of `cpu.rt_runtime_us`, which might cause it to consume too many CPU resources. In addition, this modifier ensures that rebooting a container terminates all of its processes and boots it to a clean state.
`-R`|`--release=`*major.minor*	Specifies the major release number and minor update number of the Oracle release to install. The value of *major* can be set to 4, 5, 6, or 7. If you specify `latest` for *minor*, the latest available release packages for the major release are installed. If the host is running

Oracle Linux, the default release is the same as the release installed on the host. Otherwise, the default release is the latest update of Oracle Linux 6.

`-r` | `--rpms=`*rpm_name*	Install the specified RPM in the container.
`-t` | `--templatefs=`*rootfs*	Specifies the path to the root file system of an existing system, container, or Oracle VM template that you want to copy. Do not specify this option with any other template option. See Section 28.4, "Creating Additional Containers".
`-u` | `--url=`*repo_URL*	Specifies a yum repository other than Oracle Public Yum. For example, you might want to perform the installation from a local yum server. The repository file in configured in `/etc/yum.repos.d` in the container's root file system. The default URL is `http://yum.oracle.com`.

2. If you want to create additional copies of the container in its initial state, create a snapshot of the container's root file system, for example:

```
# btrfs subvolume snapshot /container/ol6ctr1/rootfs /container/ol6ctr1/rootfs_snap
```

See Section 21.2, "About the Btrfs File System" and Section 28.4, "Creating Additional Containers".

3. Start the container `ol6ctr1` as a daemon that writes its diagnostic output to a log file other than the default log file.

```
[root@host ~]# lxc-start -n ol6ctr1 -d -o /container/ol6ctr1_debug.log -l DEBUG
```

> **Note**
>
> If you omit the `-d` option, the container's console opens in the current shell.
>
> The following logging levels are available: `FATAL`, `CRIT`, `WARN`, `ERROR`, `NOTICE`, `INFO`, and `DEBUG`. You can set a logging level for all `lxc-*` commands.

If you run the `ps -ef --forest` command on the host system and the process tree below the `lxc-start` process shows that the `/usr/sbin/sshd` and `/sbin/mingetty` processes have started in the container, you can log in to the container from the host. See Section 28.3, "Logging in to Containers".

28.2.4 About the lxc-oracle Template Script

> **Note**
>
> If you amend a template script, you alter the configuration files of all containers that you subsequently create from that script. If you amend the `config` file for a container, you alter the configuration of that container and all containers that you subsequently clone from it.

The `lxc-oracle` template script defines system settings and resources that are assigned to a running container, including:

- the default passwords for the `oracle` and `root` users, which are set to `oracle` and `root` respectively

- the host name (`lxc.utsname`), which is set to the name of the container

- the number of available terminals (`lxc.tty`), which is set to 4

- the location of the container's root file system on the host (`lxc.rootfs`)

- the location of the `fstab` mount configuration file (`lxc.mount`)

- all system capabilities that are not available to the container (`lxc.cap.drop`)

- the local network interface configuration (`lxc.network`)

- all whitelisted cgroup devices (`lxc.cgroup.devices.allow`)

The template script sets the virtual network type (`lxc.network.type`) and bridge (`lxc.network.link`) to `veth` and `virbr0`. If you want to use a macvlan bridge or Virtual Ethernet Port Aggregator that allows external systems to access your container via the network, you must modify the container's configuration file. See Section 28.2.5, "About Veth and Macvlan" and Section 28.2.6, "Modifying a Container to Use Macvlan".

To enhance security, you can uncomment `lxc.cap.drop` capabilities to prevent `root` in the container from performing certain actions. For example, dropping the `sys_admin` capability prevents `root` from remounting the container's `fstab` entries as writable. However, dropping `sys_admin` also prevents the container from mounting any file system and disables the `hostname` command. By default, the template script drops the following capabilities: `mac_admin`, `mac_override`, `setfcap`, `setpcap`, `sys_module`, `sys_nice`, `sys_pacct`, `sys_rawio`, and `sys_time`.

For more information, see the `capabilities(7)` and `lxc.conf(5)` manual pages.

When you create a container, the template script writes the container's configuration settings and mount configuration to `/container/name/config` and `/container/name/fstab`, and sets up the container's root file system under `/container/name/rootfs`.

Unless you specify to clone an existing root file system, the template script installs the following packages under `rootfs` (by default, from the Oracle Linux Yum Server at `http://yum.oracle.com`):

| Package	Description
chkconfig	`chkconfig` utility for maintaining the `/etc/rc*.d` hierarchy.
dhclient	DHCP client daemon (`dhclient`) and `dhclient-script`.
initscripts	`/etc/inittab` file and `/etc/init.d` scripts.
openssh-server	Open source SSH server daemon, `/usr/sbin/sshd`.
oraclelinux-release	Oracle Linux release and information files.
passwd	`passwd` utility for setting or changing passwords using PAM.
policycoreutils	SELinux policy core utilities.
rootfiles	Basic files required by the `root` user.
rsyslog	Enhanced system logging and kernel message trapping daemons.
vim-minimal	Minimal version of the VIM editor.
yum	`yum` utility for installing, updating and managing RPM packages.

The template script edits the system configuration files under `rootfs` to set up networking in the container and to disable unnecessary services including volume management (LVM), device management (`udev`), the hardware clock, `readahead`, and the Plymouth boot system.

28.2.5 About Veth and Macvlan

By default, the `lxc-oracle` template script sets up networking by setting up a veth bridge. In this mode, a container obtains its IP address from the `dnsmasq` server that `libvirtd` runs on the private virtual bridge network (`virbr0`) between the container and the host. The host allows a container to connect to the rest of the network by using NAT rules in `iptables`, but these rules do not allow incoming connections to the container. Both the host and other containers on the veth bridge have network access to the container via the bridge.

Figure 28.1 illustrates a host system with two containers that are connected via the veth bridge `virbr0`.

Figure 28.1 Network Configuration of Containers Using a Veth Bridge
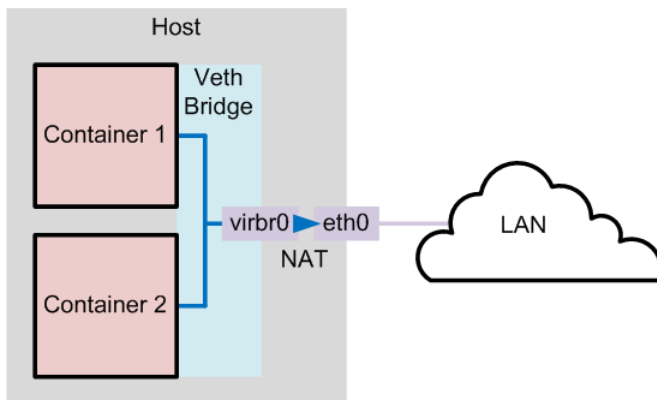
If you want to allow network connections from outside the host to be able to connect to the container, the container needs to have an IP address on the same network as the host. One way to achieve this configuration is to use a macvlan bridge to create an independent logical network for the container. This network is effectively an extension of the local network that is connected the host's network interface. External systems can access the container as though it were an independent system on the network, and the container has network access to other containers that are configured on the bridge and to external systems. The container can also obtain its IP address from an external DHCP server on your local network. However, unlike a veth bridge, the host system does not have network access to the container.

Figure 28.2 illustrates a host system with two containers that are connected via a macvlan bridge.

**Figure 28.2 Network Configuration of Containers Using a Macvlan Bridge**
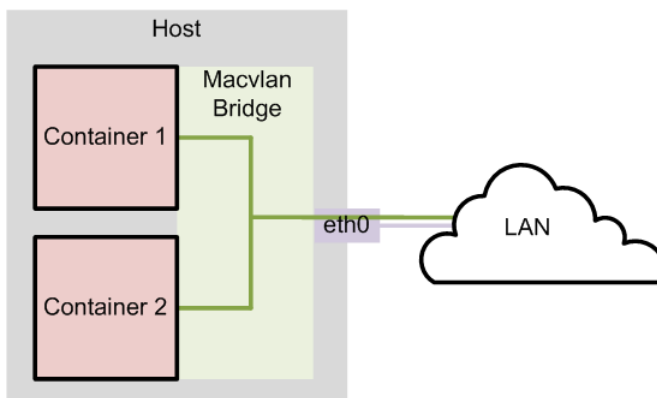
If you do not want containers to be able to see each other on the network, you can configure the Virtual Ethernet Port Aggregator (VEPA) mode of macvlan. Figure 28.3 illustrates a host system with two

containers that are separately connected to a network by a macvlan VEPA. In effect, each container is connected directly to the network, but neither container can access the other container nor the host via the network.

Figure 28.3 Network Configuration of Containers Using a Macvlan VEPA
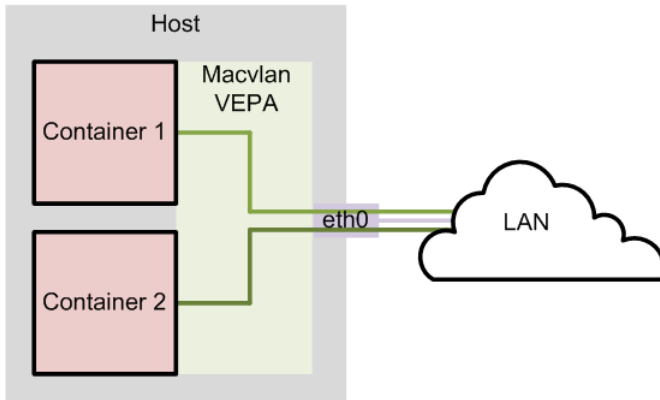
For information about configuring macvlan, see Section 28.2.6, "Modifying a Container to Use Macvlan" and the `lxc.conf(5)` manual page.

28.2.6 Modifying a Container to Use Macvlan

To modify a container so that it uses the bridge or VEPA mode of macvlan, edit `/container/`*`name`*`/config` and replace the following lines:

```
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = virbr0
```

with these lines for bridge mode:

```
lxc.network.type = macvlan
lxc.network.macvlan.mode = bridge
lxc.network.flags = up
lxc.network.link = eth0
```

or these lines for VEPA mode:

```
lxc.network.type = macvlan
lxc.network.macvlan.mode = vepa
lxc.network.flags = up
lxc.network.link = eth0
```

In these sample configurations, the setting for `lxc.network.link` assumes that you want the container's network interface to be visible on the network that is accessible via the host's `eth0` interface.

28.2.6.1 Modifying a Container to Use a Static IP Address

By default, a container connected by macvlan relies on the DHCP server on your local network to obtain its IP address. If you want the container to act as a server, you would usually configure it with a static IP address. You can configure DHCP to serve a static IP address for a container or you can define the address in the container's `config` file.

To configure a static IP address that a container does not obtain using DHCP:

1. Edit `/container/`*`name`*`/rootfs/etc/sysconfig/network-scripts/ifcfg-`*`iface`*, where *`iface`* is the name of the network interface, and change the following line:

   ```
   BOOTPROTO=dhcp
   ```

   to read:

   ```
   BOOTPROTO=none
   ```

2. Add the following line to `/container/`*`name`*`/config`:

   ```
   lxc.network.ipv4 = xxx.xxx.xxx.xxx/prefix_length
   ```

   where *`xxx.xxx.xxx.xxx/prefix_length`* is the IP address of the container in CIDR format, for example: `192.168.56.100/24`.

   > **Note**
   >
   > The address must not already be in use on the network or potentially be assignable by a DHCP server to another system.
   >
   > You might also need to configure the firewall on the host to allow access to a network service that is provided by a container.

28.3 Logging in to Containers

You can use the `lxc-console` command to log in to a running container.

```
[root@host ~]# lxc-console -n name [-t tty_number]
```

If you do not specify a tty number, you log in to the first available terminal.

For example, log in to a terminal on `ol6ctr1`:

```
[root@host ~]# lxc-console -n ol6ctr1
```

To exit an `lxc-console` session, type `Ctrl-A` followed by `Q`.

Alternatively, you can use `ssh` to log in to a container if you install the `lxc-0.9.0-2.0.5` package (or later version of this package).

> **Note**
>
> To be able to log in using `lxc-console`, the container must be running an `/sbin/mingetty` process for the terminal. Similarly, using `ssh` requires that the container is running the SSH daemon (`/usr/sbin/sshd`).

28.4 Creating Additional Containers

To clone an existing container, use the `lxc-clone` command, as shown in this example:

```
[root@host ~]# lxc-clone -o ol6ctr1 -n ol6ctr2
```

Alternatively, you can use the `lxc-create` command to create a container by copying the root file system from an existing system, container, or Oracle VM template. Specify the path of the root file system as the argument to the `--templatefs` template option:

```
[root@host ~]# lxc-create -n ol6ctr3 -B btrfs -t oracle -- --templatefs=/container/ol6ctr1/rootfs_snap
```

This example copies the new container's `rootfs` from a snapshot of the `rootfs` that belongs to container `ol6ctr1`. The additional container is created in `/container/ol6ctr3` and a new `rootfs` snapshot is created in `/container/ol6ctr3/rootfs`.

> **Note**
>
> For LXC version 1.0 and later, you must specify the `-B btrfs` option if you want to use the snapshot features of btrfs. For more information, see the `lxc-create(1)` manual page.
>
> To change the host name of the container, edit the `HOSTNAME` settings in `/container/`*name*`/rootfs/etc/sysconfig/network` and `/container/`*name*`/rootfs/etc/sysconfig/network-scripts/ifcfg-`*iface*, where *iface* is the name of the network interface, such as `eth0`.

28.5 Monitoring and Shutting Down Containers

To display the containers that are configured, use the `lxc-ls` command on the host.

```
[root@host ~]# lxc-ls
ol6ctr1
ol6ctr2
```

To display the containers that are running on the host system, specify the `--active` option.

```
[root@host ~]# lxc-ls --active
ol6ctr1
```

To display the state of a container, use the `lxc-info` command on the host.

```
[root@host ~]# lxc-info -n ol6ctr1
Name:           ol6ctr1
State:          RUNNING
PID:            5662
IP:             192.168.122.188
CPU use:        1.63 seconds
BlkIO use:      18.95 MiB
Memory use:     11.53 MiB
KMem use:       0 bytes
Link:           vethJHU5OA
 TX bytes:      1.42 KiB
 RX bytes:      6.29 KiB
 Total bytes:   7.71 KiB
```

A container can be in one of the following states: `ABORTING`, `RUNNING`, `STARTING`, `STOPPED`, or `STOPPING`. Although `lxc-info` might show your container to be in the `RUNNING` state, you cannot log in to it unless the `/usr/sbin/sshd` or `/sbin/mingetty` processes have started running in the container. You must allow time for the `init` or `systemd` process in the container to first start networking and the various other services that you have configured.

To view the state of the processes in the container from the host, either run `ps -ef --forest` and look for the process tree below the `lxc-start` process or use the `lxc-attach` command to run the `ps` command in the container.

```
[root@host ~]# ps -ef --forest
UID    PID    PPID  C STIME TTY     TIME      CMD
...
root   3171      1  0 09:57 ?       00:00:00 lxc-start -n ol6ctr1 -d
root   3182   3171  0 09:57 ?       00:00:00  \_ /sbin/init
root   3441   3182  0 09:57 ?       00:00:00     \_ /sbin/dhclient -H ol6ctr1 ...
```

```
root   3464  3182  0 09:57 ?        00:00:00       \_ /sbin/rsyslogd ...
root   3493  3182  0 09:57 ?        00:00:00       \_ /usr/sbin/sshd
root   3500  3182  0 09:57 pts/5  00:00:00       \_ /sbin/mingetty ... /dev/console
root   3504  3182  0 09:57 pts/1  00:00:00       \_ /sbin/mingetty ... /dev/tty1
root   3506  3182  0 09:57 pts/2  00:00:00       \_ /sbin/mingetty ... /dev/tty2
root   3508  3182  0 09:57 pts/3  00:00:00       \_ /sbin/mingetty ... /dev/tty3
root   3510  3182  0 09:57 pts/4  00:00:00       \_ /sbin/mingetty ... /dev/tty4
...
[root@host ~]# lxc-attach -n ol6ctr1 -- /bin/ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY        STAT START   TIME COMMAND
root         1  0.0  0.1  19284  1516 ?          Ss   04:57   0:00 /sbin/init
root       202  0.0  0.0   9172   588 ?          Ss   04:57   0:00 /sbin/dhclient
root       225  0.0  0.1 245096  1332 ?          Ssl  04:57   0:00 /sbin/rsyslogd
root       252  0.0  0.1  66660  1192 ?          Ss   04:57   0:00 /usr/sbin/sshd
root       259  0.0  0.0   4116   568 lxc/console Ss+ 04:57   0:00 /sbin/mingett
root       263  0.0  0.0   4116   572 lxc/tty1 Ss+  04:57   0:00 /sbin/mingetty
root       265  0.0  0.0   4116   568 lxc/tty2 Ss+  04:57   0:00 /sbin/mingetty
root       267  0.0  0.0   4116   572 lxc/tty3 Ss+  04:57   0:00 /sbin/mingetty
root       269  0.0  0.0   4116   568 lxc/tty4 Ss+  04:57   0:00 /sbin/mingetty
root       283  0.0  0.1 110240  1144 ?          R+   04:59   0:00 /bin/ps aux
```

Tip

If a container appears not to be starting correctly, examining its process tree from the host will often reveal where the problem might lie.

If you were logged into the container, the output from the `ps -ef` command would look similar to the following.

```
[root@ol6ctr1 ~]# ps -ef
UID   PID  PPID  C STIME TTY          TIME CMD
root    1     0  0 11:54 ?        00:00:00 /sbin/init
root  193     1  0 11:54 ?        00:00:00 /sbin/dhclient -H ol6ctr1 ...
root  216     1  0 11:54 ?        00:00:00 /sbin/rsyslogd -i ...
root  258     1  0 11:54 ?        00:00:00 /usr/sbin/sshd
root  265     1  0 11:54 lxc/console 00:00:00 /sbin/mingetty ... /dev/console
root  271     1  0 11:54 lxc/tty2 00:00:00 /sbin/mingetty ... /dev/tty2
root  273     1  0 11:54 lxc/tty3 00:00:00 /sbin/mingetty ... /dev/tty3
root  275     1  0 11:54 lxc/tty4 00:00:00 /sbin/mingetty ... /dev/tty4
root  297     1  0 11:57 ?        00:00:00 login -- root
root  301   297  0 12:08 lxc/tty1 00:00:00 -bash
root  312   301  0 12:08 lxc/tty1 00:00:00 ps -ef
```

Note that the process numbers differ from those of the same processes on the host, and that they all descend from process 1, `/sbin/init`, in the container.

To suspend or resume the execution of a container, use the `lxc-freeze` and `lxc-unfreeze` commands on the host.

```
[root@host ~]# lxc-freeze -n ol6ctr1
[root@host ~]# lxc-unfreeze -n ol6ctr1
```

From the host, you can use the `lxc-stop` command with the `--nokill` option to shut down the container in an orderly manner.

```
[root@host ~]# lxc-stop --nokill -n ol6ctr1
```

Alternatively, you can run a command such as `halt` while logged in to the container.

```
[root@ol6ctr1 ~]# halt

Broadcast message from root@ol6ctr1
 (/dev/tty2) at 22:52 ...
```

```
The system is going down for halt NOW!
lxc-console: Input/output error - failed to read

[root@host ~]#
```

As shown in the example, you are returned to the shell prompt on the host.

To shut down a container by terminating its processes immediately, use `lxc-stop` with the `-k` option.

```
[root@host ~]# lxc-stop -k -n ol6ctr1
```

If you are debugging the operation of a container, this is the quickest method as you would usually destroy the container and create a new version after modifying the template script.

To monitor the state of a container, use the `lxc-monitor` command.

```
[root@host ~]# lxc-monitor -n ol6ctr1
'ol6ctr1' changed state to [STARTING]
'ol6ctr1' changed state to [RUNNING]
'ol6ctr1' changed state to [STOPPING]
'ol6ctr1' changed state to [STOPPED]
```

To wait for a container to change to a specified state, use the `lxc-wait` command.

```
lxc-wait -n $CTR -s ABORTING && lxc-wait -n $CTR -s STOPPED && \
  echo "Container $CTR terminated with an error."
```

28.6 Starting a Command Inside a Running Container

> **Note**
>
> The `lxc-attach` command is supported by UEK R3 with the `lxc-0.9.0-2.0.4` package or later.

You can use `lxc-attach` to execute an arbitrary command inside a container that is already running from outside the container, for example:

```
[root@host ~]# lxc-attach -n ol6ctr1 -- /bin/ps aux
USER       PID %CPU %MEM    VSZ    RSS TTY       STAT START    TIME COMMAND
root         1  0.0  0.1  19284   1516 ?         Ss   04:57    0:00 /sbin/init
root       202  0.0  0.0   9172    588 ?         Ss   04:57    0:00 /sbin/dhclient
root       225  0.0  0.1 245096   1332 ?         Ssl  04:57    0:00 /sbin/rsyslogd
root       252  0.0  0.1  66660   1192 ?         Ss   04:57    0:00 /usr/sbin/sshd
root       259  0.0  0.0   4116    568 lxc/console Ss+ 04:57   0:00 /sbin/mingett
root       263  0.0  0.0   4116    572 lxc/tty1 Ss+  04:57    0:00 /sbin/mingetty
root       265  0.0  0.0   4116    568 lxc/tty2 Ss+  04:57    0:00 /sbin/mingetty
root       267  0.0  0.0   4116    572 lxc/tty3 Ss+  04:57    0:00 /sbin/mingetty
root       269  0.0  0.0   4116    568 lxc/tty4 Ss+  04:57    0:00 /sbin/mingetty
root       283  0.0  0.1 110240   1144 ?         R+   04:59    0:00 /bin/ps aux
```

For more information, see the `lxc-attach(1)` manual page.

28.7 Controlling Container Resources

Linux containers use cgroups in their implementation, and you can use the `lxc-cgroup` command to control the access that a container has to system resources relative to other containers. For example, to display the CPU cores to which a container can run on, enter:

```
[root@host ~]# lxc-cgroup -n ol6ctr1 cpuset.cpus
```

```
0-7
```

To restrict a container to cores 0 and 1, you would enter a command such as the following:

```
[root@host ~]# lxc-cgroup -n ol6ctr1 cpuset.cpus 0,1
```

To change a container's share of CPU time and block I/O access, you would enter:

```
[root@host ~]# lxc-cgroup -n ol6ctr2 cpu.shares 256
[root@host ~]# lxc-cgroup -n ol6ctr2 blkio.weight 500
```

Limit a container to 256 MB of memory when the system detects memory contention or low memory; otherwise, set a hard limit of 512 MB:

```
[root@host ~]# lxc-cgroup -n ol6ctr2 memory.soft_limit_in_bytes 268435456
[root@host ~]# lxc-cgroup -n ol6ctr2 memory.limit_in_bytes 53687091
```

To make the changes to a container's configuration permanent, add the settings to the file `/container/name/config`, for example:

```
# Permanently tweaked resource settings
lxc.cgroup.cpu.shares=256
lxc.cgroup.blkio.weight=500
```

For more information about the resources that can be controlled, see http://www.kernel.org/doc/Documentation/cgroups/.

28.8 Configuring ulimit Settings for an Oracle Linux Container

A container's `ulimit` setting honors the values of `ulimit` settings such as `memlock` and `nofile` in the container's version of `/etc/security/limits.conf/` provided that these values are lower than or equal to the values on the host system.

The values of `memlock` and `nofile` determine the maximum amount of address space in kilobytes that can be locked into memory by a user process and the maximum number of file descriptors that a user process can have open at the same time.

If you require a higher `ulimit` value for a container, increase the value of the settings in `/etc/security/limits.conf` on the host, for example:

```
#<domain>        <type>  <item>          <value>
*               soft    memlock         1048576
*               hard    memlock         2097152
*               soft    nofile          5120
*               hard    nofile          10240
```

A process can use the `ulimit` built-in shell command or the `setrlimit()` system call to raise the current limit for a shell above the soft limit. However, the new value cannot exceed the hard limit unless the process is owned by `root`.

You can use `ulimit` to set or display the current soft and hard values on the host or from inside the container, for example:

```
[root@host ~]# echo "host: nofile = $(ulimit -n)"
host: nofile = 1024
[root@host ~]# echo "host: nofile = $(ulimit -H -n)"
host: nofile = 4096
[root@host ~]# ulimit -n 2048
```

```
[root@host ~]# echo "host: nofile = $(ulimit -n)"
host: nofile = 2048
[root@host ~]# lxc-attach -n ol6ctr1 -- echo "container: nofile = $(ulimit -n)"
container: nofile = 1024
```

> **Note**
>
> Log out and log in again or, if possible, reboot the host before starting the container in a shell that uses the new soft and hard values for `ulimit`.

28.9 Configuring Kernel Parameter Settings for Oracle Linux Containers

If you specify the `--privileged` option with the `lxc-oracle` template script, you can adjust the values of certain kernel parameters for a container under the `/proc` hierarchy.

The container mounts `/proc` read-only with the following exceptions, which are writable:

- `/proc/sys/kernel/msgmax`

- `/proc/sys/kernel/msgmnb`

- `/proc/sys/kernel/msgmni`

- `/proc/sys/kernel/sem`

- `/proc/sys/kernel/shmall`

- `/proc/sys/kernel/shmmax`

- `/proc/sys/kernel/shmmni`

- `/proc/sys/net/ipv4/conf/default/accept_source_route`

- `/proc/sys/net/ipv4/conf/default/rp_filter`

- `/proc/sys/net/ipv4/ip_forward`

Each of these parameters can have a different value than that configured for the host system and for other containers running on the host system. The default value is derived from the template when you create the container. Oracle recommends that you change a setting only if an application requires a value other than the default value.

> **Note**
>
> Prior to UEK R3 QU6, the following host-only parameters were not visible within the container due to kernel limitations:
>
> - `/proc/sys/net/core/rmem_default`
>
> - `/proc/sys/net/core/rmem_max`
>
> - `/proc/sys/net/core/wmem_default`
>
> - `/proc/sys/net/core/wmem_max`
>
> - `/proc/sys/net/ipv4/ip_local_port_range`

- `/proc/sys/net/ipv4/tcp_syncookies`

  With UEK R3 QU6 and later, these parameters are read-only within the container to allow Oracle Database and other applications to be installed. You can change the values of these parameters only from the host. Any changes that you make to host-only parameters apply to all containers on the host.

28.10 Deleting Containers

To delete a container and its snapshot, use the `lxc-destroy` command as shown in the following example.

```
[root@host ~]# lxc-destroy -n ol6ctr2
Delete subvolume '/container/ol6ctr2/rootfs'
```

This command also deletes the `rootfs` subvolume.

28.11 Running Application Containers

You can use the `lxc-execute` command to create a temporary application container in which you can run a command that is effectively isolated from the rest of the system. For example, the following command creates an application container named `guest` that runs `sleep` for 100 seconds.

```
[root@host ~]# lxc-execute -n guest -- sleep 100
```

While the container is active, you can monitor it by running commands such as `lxc-ls --active` and `lxc-info -n guest` from another window.

```
[root@host ~]# lxc-ls --active
guest
[root@host ~]# lxc-info -n guest
Name:           guest
State:          RUNNING
PID:            11220
CPU use:        0.02 seconds
BlkIO use:      0 bytes
Memory use:     544.00 KiB
KMem use:       0 bytes
```

If you need to customize an application container, you can use a configuration file. For example, you might want to change the container's network configuration or the system directories that it mounts.

The following example shows settings from a sample configuration file where the `rootfs` is mostly not shared except for mount entries to ensure that `init.lxc` and certain library and binary directory paths are available.

```
lxc.utsname = guest
lxc.tty = 1
lxc.pts = 1
lxc.rootfs = /tmp/guest/rootfs
lxc.mount.entry=/usr/lib usr/lib none ro,bind 0 0
lxc.mount.entry=/usr/lib64 usr/lib64 none ro,bind 0 0
lxc.mount.entry=/usr/bin usr/bin none ro,bind 0 0
lxc.mount.entry=/usr/sbin usr/sbin none ro,bind 0 0
lxc.cgroup.cpuset.cpus=1
```

The mount entry for `/usr/sbin` is required so that the container can access `/usr/sbin/init.lxc` on the host system.

In practice, you should limit the host system directories that an application container mounts to only those directories that the container needs to run the application.

> **Note**
>
> To avoid potential conflict with system containers, do not use the `/container` directory for application containers.

You must also configure the required directories and symbolic links under the `rootfs` directory:

```
[root@host ~]# TMPDIR=/tmp/guest/rootfs
[root@host ~]# mkdir -p $TMPDIR/usr/lib $TMPDIR/usr/lib64 \
  $TMPDIR/usr/bin $TMPDIR/usr/sbin \
  $TMPDIR/dev/pts $TMPDIR/dev/shm $TMPDIR/proc
[root@host ~]# ln -s $TMPDIR/usr/lib $TMPDIR/lib
[root@host ~]# ln -s $TMPDIR/usr/lib64 $TMPDIR/lib64
[root@host ~]# ln -s $TMPDIR/usr/bin $TMPDIR/bin
[root@host ~]# ln -s $TMPDIR/usr/sbin $TMPDIR/sbin
```

In this example, the directories include `/dev/pts`, `/dev/shm`, and `/proc` in addition to the mount point entries defined in the configuration file.

You can then use the `-f` option to specify the configuration file (`config`) to `lxc-execute`:

```
[root@host ~]# lxc-execute -n guest -f config /usr/bin/bash
bash-4.2# ps -ef
UID    PID  PPID  C STIME TTY          TIME CMD
0      1     0  0 14:17 ?        00:00:00 /usr/sbin/init.lxc -- /usr/bin/bash
0      4     1  0 14:17 ?        00:00:00 /usr/bin/bash
0      5     4  0 14:17 ?        00:00:00 ps -ef
bash-4.2# mount
/dev/sda3 on / type btrfs (rw,relatime,seclabel,space_cache)
/dev/sda3 on /usr/lib type btrfs (ro,relatime,seclabel,space_cache)
/dev/sda3 on /usr/lib64 type btrfs (ro,relatime,seclabel,space_cache)
/dev/sda3 on /usr/bin type btrfs (ro,relatime,seclabel,space_cache)
/dev/sda3 on /usr/sbin type btrfs (ro,relatime,seclabel,space_cache)
devpts on /dev/pts type devpts (rw,relatime,seclabel,gid=5,mode=620,ptmxmode=666)
proc on /proc type proc (rw,relatime)
shmfs on /dev/shm type tmpfs (rw,relatime,seclabel)
mqueue on /dev/mqueue type mqueue (rw,relatime,seclabel)
bash-4.2# ls -l /
total 16
lrwxrwxrwx.   1 0 0   7 May 21 14:03 bin -> usr/bin
drwxr-xr-x.   1 0 0  52 May 21 14:27 dev
lrwxrwxrwx.   1 0 0   7 May 21 14:03 lib -> usr/lib
lrwxrwxrwx.   1 0 0   9 May 21 14:27 lib64 -> usr/lib64
dr-xr-xr-x. 230 0 0   0 May 21 14:27 proc
lrwxrwxrwx.   1 0 0   8 May 21 14:03 sbin -> usr/sbin
drwxr-xr-x.   1 0 0  30 May 21 12:58 usr
bash-4.2# touch /bin/foo
touch: cannot touch '/bin/foo': Read-only file system
bash-4.2# echo $?
1
```

In this example, running the `ps` command reveals that `bash` runs as a child of `init.lxc`. `mount` shows the individual directories that the container mounts read-only, such as `/usr/lib`, and `ls -l /` displays the symbolic links that you set up in `rootfs`.

Attempting to write to the read-only `/bin` file system results in an error. If you were to run the same `lxc-execute` command without specifying the configuration file, it would make the entire root file system of the host available to the container in read/write mode.

As for system containers, you can set `cgroup` entries in the configuration file and use the `lxc-cgroup` command to control the system resources to which an application container has access.

> **Note**
>
> `lxc-execute` is intended to run application containers that share the host's root file system, and not to run system containers that you create using `lxc-create`. Use `lxc-start` to run system containers.

For more information, see the `lxc-execute(1)` and `lxc.conf(5)` manual pages.

28.12 For More Information About Linux Containers

For more information, see https://wiki.archlinux.org/index.php/Linux_Containers and the LXC manual pages.